```
; ****************************************************************************
; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.10
; ----------------------------------------------------------------------------
; Last Update: 28/12/2025 (Previous: 11/08/2025)
; ----------------------------------------------------------------------------
; 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
COREequ 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     05H     ; 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     0BH             ; STATUS REGISTER B  ALARM
CMOS_REG_C  EQU     0CH             ; 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
                                
                                ; 04/12/2023 - TRDOS 386 v2.0.7
                                SYSTEMSTACK_ADDR  equ 97000h      ; max. 3072 bytes (96400h-97000h)
                                VBE3BIOSCODE_ADDR equ 60000h      ; Protected Mode Video Bios (64KB)
                                ;AC97DMABUFFR_ADDR equ 40000h     ; AC97 & VIA VT8233 DMA Buffer (128KB)
                                SB16DMABUFFR_ADDR equ 50000h      ; Sound Blaster 16 DMA Buffer (64KB)
                                sb16_dma_buffer equ SB16DMABUFFR_ADDR
                                
                                ; 29/08/2023 - TRDOS 386 v2.0.6
                                    ; 30/11/2020
                                    ; 29/11/2020 - TRDOS 386 v2.0.3
                                
                                struc PMInfo  ;  VESA VBE3 PMInfoBlock ('PMID' block)
                                
00000000 ????????                .Signature:       resb 4  ; db 'PMID' ; PM Info Block Signature
00000004 ????                    .EntryPoint:      resw 1  ; Offset of PM entry point within BIOS
00000006 ????                    .PMInitialize: resw 1     ; Offset of PM initialization entry point
00000008 ????                    .BIOSDataSel:     resw 1  ; Selector to BIOS data area emulation block
0000000A ????                    .A0000Sel: resw 1  ; Selector to access A0000h physical mem
0000000C ????                    .B0000Sel: resw 1  ; Selector to access B0000h physical mem
0000000E ????                    .B8000Sel: resw 1  ; Selector to access B8000h physical mem
00000010 ????                    .CodeSegSel:      resw 1  ; Selector to access code segment as data
00000012 ??                      .InProtectMode: resb 1 ; Set to 1 when in protected mode
00000013 ??                      .Checksum: resb 1  ; Checksum byte for structure
                                 .size:
                                
                                endstruc
                                
                                [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[7866]                 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[7466]                 mov     [mem_1m_1k], cx
00000017 8916[7666]                 mov     [mem_16m_64k], dx
                                    ; 24/11/2023
0000001B 8916[180F]                 mov     [real_mem_16m_64k], dx
                                
                                    ; 05/11/2014
                                    ;and    dx, dx
                                    ;jz     short L2
0000001F 81F90004                      cmp     cx, 1024
                                    ;jnb    short L0
00000023 7351                       jnb     short V0 ; 14/11/2020
                                             ; insufficient memory_error
                                             ; Minimum 2 MB memory is needed...
                                    ; 05/11/2014
                                    ; (real mode error printing)
00000025 FB                         sti
```

```
00000026 BE[3A00]                   mov     si, msg_out_of_memory
00000029 BB0700                     mov     bx, 7
0000002C B40E                       mov     ah, 0Eh ; write tty
                                oom_1:
0000002E AC                         lodsb
0000002F 08C0                       or      al, al
00000031 7404                       jz      short oom_2
00000033 CD10                       int     10h
00000035 EBF7                       jmp     short oom_1
                                oom_2:
00000037 F4                             hlt
00000038 EBFD                       jmp     short oom_2

                                ; 20/02/2017
                                ; 05/11/2014
                                msg_out_of_memory:
0000003A 070D0A                     db      07h, 0Dh, 0Ah
0000003D 496E73756666696369-            db      'Insufficient memory !'
00000046 656E74206D656D6F72-
0000004F 792021
00000052 0D0A                       db      0Dh, 0Ah
                                _int13h_48h_buffer: ; 07/07/2016
00000054 284D696E696D756D20-        db      '(Minimum 2MB memory is needed.)'
0000005D 324D42206D656D6F72-
00000066 79206973206E656564-
0000006F 65642E29
00000073 0D0A00                     db      0Dh, 0Ah, 0
                                v0:
                                    ; 18/10/2023 - TRDOS 386 v2.0.7
                                    ; set video mode to 03h again
                                    ; (to reset video bios data in ROMBIOS DATA AREA)
00000076 B80300                     mov     ax, 3
00000079 CD10                       int     10h
                                    ; copy IVT and ROMBIOS DATA AREA to VBE3 BIOS data area
0000007B BF0097                     mov     di, VBE3BIOSDATABLOCK>>4
0000007E 8EC7                       mov     es, di
00000080 31F6                       xor     si, si
00000082 31FF                       xor     di, di
00000084 8EDE                       mov     ds, si ; 0
00000086 B90003                     mov     cx, 300h ; 600h / 2
00000089 F3A5                       rep     movsw
0000008B 0E                         push    cs
0000008C 1F                         pop     ds

                                    ; 24/11/2023
                                    ; 15/12/2020
                                    ;mov    si, [mem_16m_64k]
                                    ;mov    [real_mem_16m_64k], si

                                    ; 15/11/2020
                                    ; 14/11/2020 (TRDOS 386 v2.0.3)
                                    ; check VESA (VBE) VIDEO BIOS version

0000008D B8034F                     mov     ax, 4F03h  ; Return current VBE mode
00000090 CD10                       int     10h
00000092 83F84F                     cmp     ax, 004Fh  ; successful (vbe) function call
                                    ;jne    short L0   ; not a VESA VBE compatible bios
                                    ; 18/10/2023
00000095 7561                       jne     short V1   ; restore es

                                    ; 27/11/2023
                                    ; 24/11/2023 - temporary
                                    ;jmp    short V1

                                    ;mov    ah, 3
                                    ;;jmp   short V1

                                    ; 15/11/2020
00000097 BBE097                     mov     bx, VBE3INFOSEG  ; 97E0h for current version
0000009A 8EC3                       mov     es, bx
0000009C 31FF                       xor     di, di
0000009E 2666C70556424532           mov     dword [es:di], 'VBE2' ; request VESA VBE3 info
                                            ; es:di = buffer address (512 bytes)
                                    ;mov    ax, 4F00h ; Return VBE controller information
000000A6 86E0                       xchg    al, ah
000000A8 CD10                       int     10h

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

000000AA 83F84F                     cmp     ax, 004Fh
000000AD 7549                       jne     short V1 ; old vga bios (not VESA compatible)

                                    ; 15/11/2020
000000AF 2666813D56455341           cmp     dword [es:di], 'VESA'
000000B7 753F                       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
000000B9 268A4505                   mov     al, [es:di+5]
                                            ; al = high byte of VBE version number (02h or 03h)

000000BD A2[8609]                   mov     [vbe3], al ; version number (major)
                                                       ; 02h or 03h is expected
                                    ; 17/01/2021
                                    ; Read EDID
000000C0 B301                       mov     bl, 01h ; Read EDID
000000C2 31C9                       xor     cx, cx  ; Controller unit number
                                                    ; (00 = primary controller)
000000C4 31D2                       xor     dx, dx  ; EDID block number = 0
000000C6 B8C097                     mov     ax, VBE3MODEINFOSEG  ; 97C0h for current version
000000C9 8EC0                       mov     es, ax
000000CB BF0001                     mov     di, VBE3EDIDINFOBLOCK - VBE3MODEINFOBLOCK
                                    ; es:di = temporary address of 128 bytes EDID information
000000CE B8154F                     mov     ax, 4F15h ; VBE/DDC Services
000000D1 CD10                       int     10h
                                    ;cmp    ax, 4Fh
                                    ;jne    short v2
000000D3 A2[1942]                   mov     [edid], al ; 4Fh > 0
                                ;v2:
                                    ; 17/01/2021
000000D6 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
000000D8 B91841                     mov     cx, 04118h  ; 1024*768, 24 bpp, LFB
```

```
000000DB B8014F                     mov     ax, 4F01h ; Return VBE mode information
000000DE 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
000000E0 268B752A                   mov     si, [es:di+MODEINFO.PhysBasePtr] ; hw of LFB addr
000000E4 8936[1A0F]                 mov     [def_LFB_addr], si ; k_LFB_size = 3145728 bytes
000000E8 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)
000000EC 8B3E[7666]                 mov     di, [mem_16m_64k]
                                    ; 24/11/2023
                                    ;mov    [real_mem_16m_64k], di

000000F0 39F7                       cmp     di, si
000000F2 7604                       jna     short V1

000000F4 8936[7666]                 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:
000000F8 1E                             push    ds
000000F9 07                             pop     es ; restore extra data segment
                                L0:

                                    %include 'diskinit.s' ; 07/03/2015
                                <1> ; ****************************************************************************
                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.5 - diskinit.s
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Last Update: 09/08/2022 (Previous: 29/08/2020 - Kernel v2.0.4)
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Beginning: 24/01/2016
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Assembler: NASM version 2.15 (trdos386.s)
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Turkish Rational DOS
                                <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
                                <1> ;
                                <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
                                <1> ; 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: 12/07/2022 (Previous: 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>     ; 09/08/2022
                                <1>     ; 08/08/2022
                                <1>     ; 14/07/2022 (TRDOS 386 v2.0.5)
                                <1>     ; 12/07/2022 (Retro UNIX 386 v1.2)
                                <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)
000000FA BA7F00                 <1>     mov     dx, 7Fh
                                <1> L1:
000000FD FEC2                   <1>     inc     dl
000000FF B441                   <1>     mov     ah, 41h ; Check extensions present
                                <1>                     ; Phoenix EDD v1.1 - EDD v3
00000101 BBAA55                 <1>     mov     bx, 55AAh
00000104 CD13                   <1>     int     13h
00000106 721A                   <1>     jc      short L2
                                <1>
00000108 81FB55AA               <1>     cmp     bx, 0AA55h
0000010C 7514                   <1>     jne     short L2
0000010E FE06[7B66]             <1>     inc     byte [hdc]      ; count of hard disks (EDD present)
00000112 8816[7A66]             <1>         mov     [last_drv], dl  ; last hard disk number
00000116 BB[FE65]               <1>     mov     bx, hd0_type - 80h
00000119 01D3                   <1>     add     bx, dx
0000011B 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
0000011D 80FA83                 <1>     cmp     dl, 83h  ; drive number < 83h
00000120 72DB                   <1>     jb      short L1
                                <1> L2:
                                <1>     ; 23/11/2014
                                <1>     ; 19/11/2014
00000122 30D2                   <1>     xor     dl, dl  ; 0
                                <1>     ; 04/02/2016 (esi -> si)
00000124 BE[7C66]               <1>     mov     si, fd0_type
```

```
                        <1> L3:
                        <1>     ; 14/01/2015
00000127 8816[7966]     <1>     mov     [drv], dl
                        <1>     ;
0000012B B408           <1>     mov     ah, 08h ; Return drive parameters
0000012D CD13           <1>     int     13h
0000012F 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>             ;
00000131 881C           <1>     mov     [si], bl ;  Drive type
                        <1>                      ; 4 = 1.44 MB, 80 track, 3 1/2"
                        <1>     ; 14/01/2015
00000133 E8DD01         <1>     call    set_disk_parms
                        <1>     ; 10/12/2014
00000136 81FE[7C66]     <1>     cmp     si, fd0_type
0000013A 7705           <1>     ja      short L4
0000013C 46             <1>     inc     si ; fd1_type
0000013D B201           <1>     mov     dl, 1
0000013F EBE6           <1>     jmp     short L3
                        <1> L4:
00000141 B27F           <1>     mov     dl, 7Fh
                        <1>     ; 24/12/2014
00000143 803E[7B66]00   <1>     cmp     byte [hdc], 0 ; EDD present or not ?
                        <1>     ;ja     L10       ; yes, all fixed disk operations
                        <1>                       ; will be performed according to
                        <1>                       ; present EDD specification
                        <1>     ; 14/07/2022
00000148 7603           <1>     jna     short L5
0000014A E98B00         <1>     jmp     L10
                        <1>
                        <1> L5:
                        <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>
0000014D FEC2           <1>     inc     dl
0000014F 8816[7966]     <1>         mov     [drv], dl
00000153 8816[7A66]     <1>         mov     [last_drv], dl ; 14/01/2015
00000157 B408           <1>     mov     ah, 08h ; Return drive parameters
00000159 CD13           <1>     int     13h     ; (conventional function)
                        <1>     ;jc     L13     ; fixed disk drive not ready
                        <1>     ; 14/07/2022
0000015B 7303           <1>     jnc     short L6
0000015D E9A501         <1>     jmp     L13
                        <1> L6:
00000160 8816[7B66]     <1>         mov     [hdc], dl ; number of drives
                        <1>     ;; 14/01/2013
                        <1>     ;;push  cx
00000164 E8AC01         <1>     call    set_disk_parms
                        <1>     ;;pop   cx
                        <1>     ;
                        <1>     ;;and   cl, 3Fh  ; sectors per track (bits 0-6)
00000167 8A16[7966]     <1>         mov     dl, [drv]
0000016B BB0401         <1>     mov     bx, 65*4 ; hd0 parameters table (INT 41h)
0000016E 80FA80         <1>     cmp     dl, 80h
00000171 7603           <1>     jna     short L7
00000173 83C314         <1>     add     bx, 5*4  ; hd1 parameters table (INT 46h)
                        <1> L7:
00000176 31C0           <1>     xor     ax, ax
00000178 8ED8           <1>     mov     ds, ax
0000017A 8B37           <1>         mov     si, [bx]
0000017C 8B4702         <1>         mov     ax, [bx+2]
0000017F 8ED8           <1>     mov     ds, ax
00000181 3A4C0E         <1>         cmp     cl, [si+FDPT_SPT] ; sectors per track
                        <1>     ;jne    L12 ; invalid FDPT
                        <1>     ; 14/07/2022
00000184 7403           <1>     je      short L7_8
00000186 E97801         <1>     jmp     L12
                        <1> L7_8:
00000189 BF0000         <1>     mov     di, HD0_DPT
0000018C 80FA80         <1>     cmp     dl, 80h
0000018F 7603           <1>     jna     short L8
00000191 BF2000         <1>     mov     di, HD1_DPT
                        <1> L8:
                        <1>     ; 30/12/2014
00000194 B80090         <1>     mov     ax, DPT_SEGM
00000197 8EC0           <1>     mov     es, ax
                        <1>     ; 24/12/2014
00000199 B90800         <1>     mov     cx, 8
0000019C F3A5           <1>     rep     movsw  ; copy 16 bytes to the kernel's DPT location
0000019E 8CC8           <1>     mov     ax, cs
000001A0 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)
000001A2 B8F001         <1>     mov     ax, 1F0h
000001A5 B3A0           <1>     mov     bl, 0A0h
000001A7 80FA80         <1>     cmp     dl, 80h
000001AA 7603           <1>     jna     short L9
                        <1>     ; dl = 81h
000001AC 80C310         <1>     add     bl, 10h  ; slave disk
                        <1>     ;sub    ax, 1F0h-170h
                        <1> L9:
000001AF AB             <1>     stosw  ; I/O PORT Base Address (1F0h, 170h)
000001B0 050602         <1>     add     ax, 206h
000001B3 AB             <1>     stosw  ; CONTROL PORT Address (3F6h, 376h)
000001B4 88D8           <1>     mov     al, bl  ; bit 4, master/slave disk bit
                        <1>     ;add    al, 0A0h ; 17/07/2020
000001B6 AA             <1>     stosb  ; Device/Head Register upper nibble
                        <1>     ;
000001B7 FE06[7966]     <1>     inc     byte [drv]
                        <1>     ;mov    bx, hd0_type - 80h
                        <1>     ;add    bx, cx
                        <1>     ; 09/08/2022 - BugFix
000001BB 30FF           <1>     xor     bh, bh
000001BD 88D3           <1>     mov     bl, dl
000001BF 81C3[FE65]     <1>     add     bx, hd0_type - 80h
000001C3 800F80         <1>     or      byte [bx], 80h  ; present sign (when lower nibble is 0)
000001C6 A0[7B66]       <1>     mov     al, [hdc]
```

```
000001C9 FEC8                <1>     dec     al
                             <1>     ;jz     L13
                             <1>     ; 14/07/2022
000001CB 7408                <1>     jz      short L9_10
000001CD 80FA80              <1>     cmp     dl, 80h
                             <1>         ;jna       L5 ; Max. 2 hard disks  ; 17/07/2020
                             <1>     ; 14/07/2022
000001D0 7703                <1>     ja      short L9_10
000001D2 E978FF              <1>     jmp     L5
                             <1> L9_10:
000001D5 E92D01              <1>         jmp     L13
                             <1> L10:
000001D8 FEC2                <1>     inc     dl
                             <1>     ; 25/12/2014
000001DA 8816[7966]          <1>     mov     [drv], dl
000001DE B408                <1>     mov     ah, 08h ; Return drive parameters
000001E0 CD13                <1>     int     13h     ; (conventional function)
                             <1>     ;jc     L13
                             <1>     ; 14/07/2022
000001E2 72F1                <1>     jc      short L9_10
                             <1>     ; 14/01/2015
                             <1>     ;mov    dl, [drv]
                             <1>     ; 09/08/2022
                             <1>     ;push   dx
000001E4 51                  <1>     push    cx
000001E5 E82B01              <1>     call    set_disk_parms
000001E8 59                  <1>     pop     cx
                             <1>     ;pop    dx
                             <1>     ; 09/08/2022
000001E9 8A16[7966]          <1>     mov     dl, [drv]
                             <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
000001ED BE[5400]            <1>     mov     si, _int13h_48h_buffer
                             <1>     ; 09/07/2016
000001F0 B81E00              <1>     mov     ax, 001Eh
000001F3 8824                <1>     mov     [si], ah ; 0
000001F5 46                  <1>     inc     si
000001F6 8904                <1>     mov     [si], ax
                             <1>     ; word [si] = 30
                             <1>     ;
000001F8 B448                <1>     mov     ah, 48h ; Get drive parameters (EDD function)
000001FA CD13                <1>     int     13h
                             <1>     ;jc     L13
                             <1>     ; 14/07/2022
000001FC 72D7                <1>     jc      short L9_10
                             <1>
                             <1>     ; 29/08/2020
                             <1>     ; 04/02/2016 (ebx -> bx)
                             <1>     ; 14/01/2015
000001FE 28FF                <1>     sub     bh, bh
00000200 88D3                <1>     mov     bl, dl
                             <1>     ;sub    bl, 80h
                             <1>     ; 29/08/2020
00000202 81C3[FE65]          <1>     add     bx, (hd0_type - 80h)
                             <1>     ;mov    al, [bx]
00000206 8A07                <1>     mov     al, [bx]
00000208 0C80                <1>     or      al, 80h
0000020A 8807                <1>     mov     [bx], al
0000020C 81EB[7C66]          <1>     sub     bx, hd0_type - 2 ; 15/01/2015
                             <1>     ;add    bx, drv.status
                             <1>     ;mov    [bx], al
                             <1>     ; 29/08/2020
00000210 8887[9E66]          <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
00000214 8B4410              <1>     mov     ax, [si+16]
00000217 8B7C12              <1>     mov     di, [si+18]
0000021A 09C0                <1>     or      ax, ax
0000021C 7504                <1>     jnz     short L10_LBA
0000021E 09FF                <1>     or      di, di
00000220 740B                <1>     jz      short L10_A0h
                             <1> L10_LBA:
                             <1>     ;sub    bx, drv.status
00000222 C1E302              <1>     shl     bx, 2
                             <1>     ;add    bx, drv.size ; disk size (in sectors)
                             <1>     ;mov    [bx], ax
                             <1>     ; 29/08/2020
00000225 8987[8266]          <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
00000229 89BF[8466]          <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
0000022D BF0000              <1>     mov     di, HD0_DPT
00000230 88D0                <1>     mov     al, dl
00000232 83E003              <1>     and     ax, 3
00000235 C0E005              <1>     shl     al, 5 ; * 32
00000238 01C7                <1>     add     di, ax
0000023A B80090              <1>     mov     ax, DPT_SEGM
0000023D 8EC0                <1>     mov     es, ax
                             <1>     ;
0000023F 88E8                <1>     mov     al, ch  ; max. cylinder number (bits 0-7)
00000241 88CC                <1>     mov     ah, cl
00000243 C0EC06              <1>     shr     ah, 6   ; max. cylinder number (bits 8-9)
00000246 40                  <1>     inc     ax      ; logical cylinders (limit 1024)
00000247 AB                  <1>     stosw
00000248 88F0                <1>     mov     al, dh  ; max. head number
                             <1>     ;
0000024A 30F6                <1>     xor     dh, dh  ; 29/08/2020 (dh = 0 is needed here)
```

```
                                <1>     ;
0000024C FEC0                   <1>     inc     al
0000024E AA                     <1>     stosb           ; logical heads (limits 256)
0000024F B0A0                   <1>     mov     al, 0A0h ; Indicates translated table
00000251 AA                     <1>     stosb
00000252 8A440C                 <1>     mov     al, [si+12]
00000255 AA                     <1>     stosb           ; physical sectors per track
00000256 31C0                   <1>     xor     ax, ax
                                <1>     ;dec    ax      ; 02/01/2015
00000258 AB                     <1>     stosw           ; precompensation (obsolete)
                                <1>     ;xor    al, al  ; 02/01/2015
00000259 AA                     <1>     stosb           ; reserved
0000025A B008                   <1>     mov     al, 8   ; drive control byte
                                <1>                     ; (do not disable retries,
                                <1>                     ; more than 8 heads)
0000025C AA                     <1>     stosb
0000025D 8B4404                 <1>     mov     ax, [si+4]
00000260 AB                     <1>     stosw           ; physical number of cylinders
                                <1>     ;push   ax      ; 02/01/2015
00000261 8A4408                 <1>     mov     al, [si+8]
00000264 AA                     <1>     stosb           ; physical num. of heads (limit 16)
00000265 29C0                   <1>     sub     ax, ax
                                <1>     ;pop    ax      ; 02/01/2015
00000267 AB                     <1>     stosw           ; landing zone (obsolete)
00000268 88C8                   <1>     mov     al, cl  ; logical sectors per track (limit 63)
0000026A 243F                   <1>     and     al, 3Fh
0000026C AA                     <1>     stosb
                                <1>     ;sub    al, al  ; checksum
                                <1>     ;stosb
                                <1>     ;
0000026D 83C61A                 <1>     add     si, 26   ; (BIOS) DPTE address pointer
00000270 AD                     <1>     lodsw
00000271 50                     <1>     push    ax ; *  ; (BIOS) DPTE offset
00000272 AD                     <1>     lodsw
00000273 50                     <1>     push    ax ; ** ; (BIOS) DPTE segment
                                <1>     ;
                                <1>     ; checksum calculation
00000274 89FE                   <1>     mov     si, di
00000276 06                     <1>     push    es
00000277 1F                     <1>     pop     ds
                                <1>     ;mov    cx, 16
00000278 B90F00                 <1>     mov     cx, 15
0000027B 29CE                   <1>     sub     si, cx
0000027D 30E4                   <1>     xor     ah, ah
                                <1>     ;del    cl
                                <1> L11:
0000027F AC                     <1>     lodsb
00000280 00C4                   <1>     add     ah, al
00000282 E2FB                   <1>     loop    L11
                                <1>     ;
00000284 88E0                   <1>     mov     al, ah
00000286 F6D8                   <1>     neg     al      ; -x+x = 0
00000288 AA                     <1>     stosb           ; put checksum in byte 15 of the tbl
                                <1>     ;
00000289 1F                     <1>     pop     ds ; ** ; (BIOS) DPTE segment
0000028A 5E                     <1>     pop     si ; *  ; (BIOS) DPTE offset
                                <1>
                                <1>     ; 08/08/2022 (TRDOS 386 v2.0.5)
                                <1>     ; (Recent version of Retro UNIX 386 v1 'diskinit.s' file
                                <1>     ; -12/07/2022- does not contain following 2020 code) (*)
                                <1>
                                <1>     ; 14/07/2020 (TRDOS 386 v2.0.2)
                                <1>     ; 0FFFFh:0FFFFh = invalid DPTE address
0000028B 8B0C                   <1>     mov     cx, [si]
0000028D 8B4402                 <1>     mov     ax, [si+2]
00000290 21C1                   <1>     and     cx, ax
00000292 41                     <1>     inc     cx
00000293 7404                   <1>     jz      short L11c ; 0FFFFh:0FFFFh
00000295 0B04                   <1>     or      ax, [si]
00000297 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>
00000299 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>
0000029C B3A0                   <1>     mov     bl, 0A0h
                                <1>
0000029E F6C201                 <1>     test    dl, 1
000002A1 7403                   <1>     jz      short L11g  ; Master (as default, for 80h & 82h))
                                <1>     ;shl    bl, 4 ; bl = 16 (bit 4 = 1 -> slave)
000002A3 80C310                 <1>     add     bl, 10h  ; Slave (as default, for 81h & 83h)
                                <1> L11g:
                                <1>     ; 17/07/2020
000002A6 80FA82                 <1>     cmp     dl, 82h ; Hard disk 3 or 4 ?
000002A9 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>
000002AB 2D8000                 <1>     sub     ax, 1F0h-170h
                                <1> L11d:
                                <1>     ; 14/07/2020
000002AE AB                     <1>     stosw   ; I/O PORT Base Address (1F0h, 170h)
000002AF 050602                 <1>     add     ax, 206h
000002B2 AB                     <1>     stosw   ; CONTROL PORT Address (3F6h, 376h)
000002B3 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)
000002B5 30E4                   <1>     xor     ah, ah
                                <1>     ;stosb  ; Device/Head Register upper nibble
000002B7 AB                     <1>     stosw
000002B8 30C0                   <1>     xor     al, al
```

```
000002BA B90500              <1>     mov     cx, 5
000002BD F3AB                <1>     rep     stosw ; clear remain part of the (fake) DPTE
000002BF 0E                  <1>     push    cs
000002C0 1F                  <1>     pop     ds
000002C1 EB2E                <1>     jmp     short L11f
                             <1>
                             <1>     ; 08/08/2022 (TRDOS 386 v2.0.5)
                             <1>     ; (Recent version of Retro UNIX 386 v1 'diskinit.s' file
                             <1>     ; -12/07/2022- does not contain above 2020 code) (*)
                             <1> L11e:
                             <1>     ; 23/02/2015
000002C3 57                  <1>     push    di
                             <1>     ; ES:DI points to DPTE (FDPTE) location
                             <1>     ;;mov   cx, 8
                             <1>     ;mov    cl, 8
000002C4 B90800              <1>     mov     cx, 8 ; 14/07/2020
000002C7 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)
000002C9 5B                  <1>     pop     bx
000002CA 8CC8                <1>     mov     ax, cs
000002CC 8ED8                <1>     mov     ds, ax
000002CE 268B07              <1>     mov     ax, [es:bx]
000002D1 3DF001              <1>     cmp     ax, 1F0h
000002D4 7413                <1>     je      short L11a
000002D6 3D7001              <1>     cmp     ax, 170h
000002D9 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
000002DB 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
000002DD C687[FE65]00        <1>     mov     byte [bx+hd0_type-80h], 0
000002E2 808F[2066]F0        <1>     or      byte [bx+drv.status-7Eh], 0F0h
000002E7 EB0F                <1>     jmp     short L11b
                             <1> L11a:
                             <1>     ; LBA validation
000002E9 268A4704            <1>     mov     al, [es:bx+4] ; Head register upper nibble
000002ED A840                <1>     test    al, 40h ; LBA bit (bit 6)
000002EF 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
000002F1 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
000002F3 80A7[2066]FE        <1>     and     byte [bx+drv.status-7Eh], 0FEh
                             <1>     ; 'diskio' procedure will check this bit !
                             <1> L11b:
000002F8 3A16[7A66]          <1>     cmp     dl, [last_drv] ; 25/12/2014
000002FC 7307                <1>         jnb     short L13
000002FE E9D7FE              <1>         jmp     L10
                             <1>
                             <1> L12:
                             <1>     ; Restore data registers
00000301 8CC8                <1>     mov     ax, cs
00000303 8ED8                <1>     mov     ds, ax
                             <1> L13:
                             <1>     ; 13/12/2014
00000305 0E                  <1>     push    cs
00000306 07                  <1>     pop     es
                             <1> L14:
                             <1>     ; clear keyboard buffer
00000307 B411                <1>     mov     ah, 11h
00000309 CD16                <1>     int     16h
0000030B 7440                <1>     jz      short L16 ; no keys in keyboard buffer
0000030D B010                <1>     mov     al, 10h
0000030F CD16                <1>     int     16h
00000311 EBF4                <1>     jmp     short L14
                             <1>
                             <1> set_disk_parms:
                             <1>     ; 08/08/2022 - TRDOS 386 v2.0.5
                             <1>     ; 09/05/2022 - Retro UNIX 386 v1.2
                             <1>     ; 29/08/2020 - TRDOS 386 v2.0.2
                             <1>     ; 04/02/2016 (ebx -> bx)
                             <1>     ; 10/07/2015
                             <1>     ; 14/01/2015
                             <1>     ;push   bx
00000313 28FF                <1>     sub     bh, bh
00000315 8A1E[7966]          <1>     mov     bl, [drv]
00000319 80FB80              <1>     cmp     bl, 80h
0000031C 7203                <1>     jb      short sdp0
0000031E 80EB7E              <1>     sub     bl, 7Eh
                             <1> sdp0:
                             <1>     ;add    bx, drv.status
                             <1>     ;mov    byte [bx], 80h ; 'Present' flag
                             <1>     ; 29/08/2020
00000321 C687[9E66]80        <1>     mov     byte [bx+drv.status], 80h
                             <1>     ;
00000326 88E8                <1>     mov     al, ch ; last cylinder (bits 0-7)
00000328 88CC                <1>     mov     ah, cl ;
0000032A C0EC06              <1>     shr     ah, 6  ; last cylinder (bits 8-9)
                             <1>     ;sub    bx, drv.status
0000032D D0E3                <1>     shl     bl, 1
                             <1>     ;add    bx, drv.cylinders
0000032F 40                  <1>     inc     ax  ; convert max. cyl number to cyl count
                             <1>     ;mov    [bx], ax
                             <1>     ; 08/08/2022
                             <1>     ; 29/08/2020
                             <1>     ;mov    [bx+drv.cylinders], ax
                             <1>     ;
00000330 50                  <1>     push    ax ; ** cylinders
                             <1>     ;sub    bx, drv.cylinders
                             <1>     ;add    bx, drv.heads
00000331 30E4                <1>     xor     ah, ah
00000333 88F0                <1>     mov     al, dh ; heads
00000335 40                  <1>     inc     ax
                             <1>     ;mov    [bx], ax
                             <1>     ; 08/08/2022
                             <1>     ; 29/08/2020
```

```
                                <1>     ;mov    [bx+drv.heads], ax
                                <1>     ;sub    bx, drv.heads
                                <1>        ;add       bx, drv.spt
00000336 30ED                   <1>     xor     ch, ch
00000338 80E13F                 <1>     and     cl, 3Fh ; sectors (bits 0-6)
                                <1>     ;mov    [bx], cx
                                <1>        ; 08/08/2022
                                <1>     ; 29/08/2020
                                <1>     ;mov    [bx+drv.spt], cx
                                <1>     ;sub    bx, drv.spt
0000033B D1E3                   <1>     shl     bx, 1
                                <1>     ;add    bx, drv.size ; disk size (in sectors)
                                <1>     ; LBA size = cylinders * heads * secpertrack
0000033D F7E1                   <1>     mul     cx
0000033F 89C2                   <1>     mov     dx, ax  ; heads*spt
00000341 58                     <1>     pop     ax ; ** cylinders
                                <1>     ; 09/05/2022 (fd0&fd1 drv.size = cyls*spt*heads)
                                <1>     ;dec    ax ; 1 cylinder reserved (!?) ; (*)
00000342 F7E2                   <1>     mul     dx ; cylinders * (heads*spt)
                                <1>     ;mov    [bx], ax
                                <1>     ;mov    [bx+2], dx
                                <1>     ; 29/08/2020
00000344 8987[8266]             <1>     mov     [bx+drv.size], ax
00000348 8997[8466]             <1>     mov     [bx+drv.size+2], dx
                                <1>     ;
                                <1>     ;pop    bx
0000034C C3                     <1>     retn
                                <1>
                                <1> L16:; 28/05/2016
                                
                                        ; 10/11/2014
0000034D 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
0000034E B080                           mov     al, 80h
00000350 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
00000352 31C0                           xor     ax, ax
00000354 8EC0                           mov     es, ax ; ES = 0
                                        ;
                                        ; 04/07/2016 - TRDOS 386 (64K - 128K kernel)
00000356 31F6                           xor     si, si
00000358 31FF                           xor     di, di
                                        ;mov    cx, 16384
                                        ;rep    movsd
                                        ; 02/12/2023
0000035A B90080                         mov     cx, 32768
0000035D F3A5                           rep     movsw
                                        ;
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
                                        ;sub    si, si
                                        ;sub    di, di
                                        ;mov    cx, 16384
                                        ;rep    movsd
                                        ; 02/12/2023
                                        ; si = di = 0
0000036D B90080                         mov     cx, 32768
00000370 F3A5                           rep     movsw

                                        ; Turn off the floppy drive motor
00000372 BAF203                            mov     dx, 3F2h
00000375 EE                                out     dx, al ; 0 ; 31/12/2013

                                        ; Enable access to memory above one megabyte
                                    L18:
00000376 E464                           in      al, 64h
00000378 A802                           test    al, 2
0000037A 75FA                              jnz     short L18
0000037C B0D1                           mov     al, 0D1h        ; Write output port
0000037E E664                           out     64h, al
                                    L19:
00000380 E464                           in      al, 64h
00000382 A802                           test    al, 2
00000384 75FA                              jnz     short L19
00000386 B0DF                           mov     al, 0DFh        ; Enable A20 line
00000388 E660                           out     60h, al
                                    ;L20:
                                        ;
                                        ; Load global descriptor table register

                                           ;mov     ax, cs
                                           ;mov     ds, ax

0000038A 2E0F0116[E865]                    lgdt    [cs:gdtd]

00000390 0F20C0                            mov     eax, cr0
                                        ;or     al, 1   ; 24/07/2023
00000393 40                             inc     ax
00000394 0F22C0                         mov     cr0, eax

                                        ; Jump to 32 bit code

00000397 66                             db 66h                  ; Prefix for 32-bit
00000398 EA                             db 0EAh                 ; Opcode for far jump
00000399 [9F030000]                     dd StartPM              ; Offset to start, 32-bit
                                                                ; (1000h:StartPM = StartPM + 10000h)
0000039D 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
0000039F 66B81000                       mov ax, KDATA           ; Save data segment identifier
000003A3 8ED8                               mov ds, ax              ; Move a valid data segment into DS register
000003A5 8EC0                                   mov es, ax              ; Move data segment into ES register
000003A7 8EE0                                   mov fs, ax              ; Move data segment into FS register
000003A9 8EE8                                   mov gs, ax              ; Move data segment into GS register
000003AB 8ED0                               mov ss, ax              ; Move data segment into SS register
                                        ;mov esp, 90000h        ; Move the stack pointer to 090000h
                                        ; 04/12/2023 - TRDOS 386 v2.0.7
                                        ;mov esp, 97000h       ; 3072 bytes system stack (96400h-97000h)
000003AD BC00700900                     mov esp, SYSTEMSTACK_ADDR ; 97000h (max. 3072 bytes)
                                    
                                    clear_bss: ; Clear uninitialized data area
                                        ; 11/03/2015
000003B2 31C0                           xor     eax, eax ; 0
000003B4 B9790A0000                     mov     ecx, (bss_end - bss_start)/4
                                        ;shr    ecx, 2 ; bss section is already aligned for double words
000003B9 BF[93730100]                   mov     edi, bss_start
000003BE F3AB                           rep     stosd
                                    
                                    memory_init:
                                        ; Initialize memory allocation table and page tables
                                        ; 04/12/2023
                                        ; 29/11/2023
                                        ; 27/11/2023
                                        ; 23/11/2023 (TRDOS 386 v2.0.7)
                                        ; 24/07/2022 (TRDOS 386 v2.0.5)
                                        ; 18/04/2021 (TRDOS 386 v2.0.4)
                                        ; 16/11/2014
                                        ; 15/11/2014
                                        ; 07/11/2014
                                        ; 06/11/2014
                                        ; 05/11/2014
                                        ; 04/11/2014
                                        ; 31/10/2014 (Retro UNIX 386 v1 - Beginning)
                                        ;
                                    ;   xor     eax, eax
                                    ;   xor     ecx, ecx
000003C0 B108                           mov     cl, 8
000003C2 BF00001000                     mov     edi, MEM_ALLOC_TBL
000003C7 F3AB                           rep     stosd               ; clear Memory Allocation Table
                                                                    ; for the first 1 MB memory
                                        ;
000003C9 668B0D[74660000]               mov     cx, [mem_1m_1k]           ; Number of contiguous KB between
                                                                    ; 1 and 16 MB, max. 3C00h = 15 MB.
                                        ;shr    cx, 2               ; convert 1 KB count to 4 KB count
                                        ; 24/07/2022
000003D0 C1E902                         shr     ecx, 2
000003D3 890D[88760100]                 mov     [free_pages], ecx
000003D9 668B15[76660000]               mov     dx, [mem_16m_64k]  ; Number of contiguous 64 KB blocks
                                                                    ; between 16 MB and 4 GB.
000003E0 6609D2                         or      dx, dx
000003E3 7425                           jz      short mi_0
                                    
                                    ; 04/12/2023
                                    %if 1
                                        ; 02/12/2023 - temporary (2816MB limit)
                                        ;cmp    dx, 44800 ; 0AF00h
000003E5 6681FA009F                     cmp     dx, 40704      ; (2560MB limit)
000003EA 7604                           jna     short mi_x
                                        ;mov    dx, 44800
000003EC 66BA009F                       mov     dx, 40704
                                    mi_x:
                                    %endif
                                        ; 23/11/2023 - temporary
                                        ;and    dx, 3FFFh
                                    
000003F0 668915[76660000]               mov     [mem_16m_64k], dx
                                    
000003F7 6689D0                         mov     ax, dx
000003FA C1E004                         shl     eax, 4              ; 64 KB -> 4 KB (page count)
000003FD 0105[88760100]                 add     [free_pages], eax
00000403 0500100000                     add     eax, 4096           ; 16 MB = 4096 pages
00000408 EB06                           jmp     short mi_1
                                    mi_0:
                                        ;mov    ax, cx
                                        ; 24/07/2022
0000040A 89C8                           mov     eax, ecx
0000040C 66050001                       add     ax, 256             ; add 256 pages for the first 1 MB
                                        ;add    eax, 256
                                    mi_1:
00000410 A3[84760100]                   mov     [memory_size], eax ; Total available memory in pages
                                                                    ; 1 alloc. tbl. bit = 1 memory page
                                                                    ; 32 allocation bits = 32 mem. pages
                                        ;
00000415 05FF7F0000                     add     eax, 32767          ; 32768 memory pages per 1 M.A.T. page
0000041A 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
0000041D 66A3[98760100]                 mov     [mat_size], ax      ; Memory Alloc. Table Size in pages
00000423 C1E00C                         shl     eax, 12             ; 1 M.A.T. page = 4096 bytes
                                        ;                           ; Max. 32 M.A.T. pages (4 GB memory)
00000426 89C3                           mov     ebx, eax            ; M.A.T. size in bytes
                                        ; Set/Calculate Kernel's Page Directory Address
00000428 81C300001000                   add     ebx, MEM_ALLOC_TBL
0000042E 891D[80760100]                 mov     [k_page_dir], ebx  ; Kernel's Page Directory address
                                                                    ; just after the last M.A.T. page
                                        ;
00000434 83E804                         sub     eax, 4              ; convert M.A.T. size to offset value
00000437 A3[90760100]                   mov     [last_page], eax    ; last page offset in the M.A.T.
                                        ;                           ; (allocation status search must be
                                                                    ; stopped after here)
0000043C 31C0                           xor     eax, eax
0000043E 48                             dec     eax                 ; FFFFFFFFh (set all bits to 1)
                                        ;push   cx
                                        ; 18/04/2021
0000043F 51                             push    ecx
                                        ; ecx = 3840 ; 27/11/2023
                                        ; (Note: ecx < 3840 if the total memory is less than 16 MB)
00000440 C1E905                         shr     ecx, 5              ; convert 1 - 16 MB page count to
                                                                    ; count of 32 allocation bits
                                        ; ecx = 120 ; 27/11/2023
00000443 F3AB                           rep     stosd
                                        ;pop    cx
                                        ; 18/04/2021
00000445 59                             pop     ecx
00000446 40                             inc     eax                 ; 0
00000447 80E11F                         and     cl, 31              ; remain bits
0000044A 7412                           jz      short mi_4
0000044C 8907                           mov     [edi], eax          ; reset
                                    mi_2:
0000044E 0FAB07                         bts     [edi], eax          ; 06/11/2014
00000451 FEC9                           dec     cl
```

```
644 00000453 7404                       jz      short mi_3
645 00000455 FEC0                       inc     al
646 00000457 EBF5                       jmp     short mi_2
647                                 mi_3:
648 00000459 28C0                       sub     al, al          ; 0
649 0000045B 83C704                     add     edi, 4          ; 15/11/2014
650                                 mi_4:
651 0000045E 6609D2                     or      dx, dx          ; check 16 MB to 4 GB memory space
652 00000461 7421                       jz      short mi_6      ; max. 16 MB memory, no more...
653                                     ;
654 00000463 B900021000                 mov     ecx, MEM_ALLOC_TBL + 512 ; End of first 16 MB memory
655                                     ;
656 00000468 29F9                       sub     ecx, edi        ; displacement (to end of 16 MB)
657 0000046A 7406                       jz      short mi_5      ; jump if EDI points to
658                                                             ;         end of first 16 MB
659 0000046C D1E9                       shr     ecx, 1          ; convert to dword count
660 0000046E D1E9                       shr     ecx, 1          ; (shift 2 bits right)
661 00000470 F3AB                       rep     stosd           ; reset all bits for reserved pages
662                                                             ; (memory hole under 16 MB)
663                                 mi_5:
664 00000472 6689D1                     mov     cx, dx          ; count of 64 KB memory blocks
665 00000475 D1E9                       shr     ecx, 1          ; 1 alloc. dword per 128 KB memory
666 00000477 9C                         pushf                   ; 16/11/2014
667 00000478 48                         dec     eax             ; FFFFFFFFh (set all bits to 1)
668 00000479 F3AB                       rep     stosd
669 0000047B 40                         inc     eax             ; 0
670 0000047C 9D                         popf                    ; 16/11/2014
671 0000047D 7305                       jnc     short mi_6
672 0000047F 6648                       dec     ax              ; eax = 0000FFFFh
673 00000481 AB                         stosd
674 00000482 6640                       inc     ax              ; 0
675                                 mi_6:
676 00000484 39DF                       cmp     edi, ebx        ; check if EDI points to
677 00000486 730A                       jnb     short mi_7      ; end of memory allocation table
678                                     ;                       ; (>= MEM_ALLOC_TBL + 4906)
679 00000488 89D9                       mov     ecx, ebx        ; end of memory allocation table
680 0000048A 29F9                       sub     ecx, edi        ; convert displacement/offset
681 0000048C D1E9                       shr     ecx, 1          ; to dword count
682 0000048E D1E9                       shr     ecx, 1          ; (shift 2 bits right)
683 00000490 F3AB                       rep     stosd           ; reset all remain M.A.T. bits
684                                 mi_7:
685                                     ; Reset M.A.T. bits in M.A.T. (allocate M.A.T. pages)
686 00000492 BA00001000                 mov     edx, MEM_ALLOC_TBL
687                                     ;sub    ebx, edx        ; Mem. Alloc. Tbl. size in bytes
688                                     ;shr    ebx, 12         ; Mem. Alloc. Tbl. size in pages
689 00000497 668B0D[98760100]           mov     cx, [mat_size]  ; Mem. Alloc. Tbl. size in pages
690 0000049E 89D7                       mov     edi, edx
691 000004A0 C1EF0F                     shr     edi, 15         ; convert M.A.T. address to
692                                                             ; byte offset in M.A.T.
693                                                             ; (1 M.A.T. byte points to
694                                                             ;       32768 bytes)
695                                                             ; Note: MEM_ALLOC_TBL address
696                                                             ; must be aligned on 128 KB
697                                                             ; boundary!
698 000004A3 01D7                       add     edi, edx        ; points to M.A.T.'s itself
699                                     ; eax = 0
700 000004A5 290D[88760100]             sub     [free_pages], ecx ; 07/11/2014
701                                 mi_8:
702 000004AB 0FB307                     btr     [edi], eax      ; clear bit 0 to bit x (1 to 31)
703                                     ;dec    bl
704 000004AE FEC9                       dec     cl
705 000004B0 7404                       jz      short mi_9
706 000004B2 FEC0                       inc     al
707 000004B4 EBF5                       jmp     short mi_8
708                                 mi_9:
709                                     ;
710                                     ; Reset Kernel's Page Dir. and Page Table bits in M.A.T.
711                                     ;             (allocate pages for system page tables)
712
713                                     ; edx = MEM_ALLOC_TBL
714 000004B6 8B0D[84760100]             mov     ecx, [memory_size] ; memory size in pages (PTEs)
715 000004BC 81C1FF030000               add     ecx, 1023       ; round up (1024 PTEs per table)
716 000004C2 C1E90A                     shr     ecx, 10         ; convert memory page count to
717                                                             ; page table count (PDE count)
718                                     ;
719 000004C5 51                         push    ecx             ; (**) PDE count (<= 1024)
720                                     ;
721 000004C6 41                         inc     ecx             ; +1 for kernel page directory
722                                     ;
723 000004C7 290D[88760100]             sub     [free_pages], ecx ; 07/11/2014
724                                     ;
725 000004CD 8B35[80760100]             mov     esi, [k_page_dir] ; Kernel's Page Directory address
726 000004D3 C1EE0C                     shr     esi, 12         ; convert to page number
727                                 mi_10:
728 000004D6 89F0                       mov     eax, esi        ; allocation bit offset
729 000004D8 89C3                       mov     ebx, eax
730 000004DA C1EB03                     shr     ebx, 3          ; convert to alloc. byte offset
731 000004DD 80E3FC                     and     bl, 0FCh        ; clear bit 0 and bit 1
732                                                             ;   to align on dword boundary
733 000004E0 83E01F                     and     eax, 31         ; set allocation bit position
734                                                             ;  (bit 0 to bit 31)
735                                     ;
736 000004E3 01D3                       add     ebx, edx        ; offset in M.A.T. + M.A.T. address
737                                     ;
738 000004E5 0FB303                     btr     [ebx], eax      ; reset relevant bit (0 to 31)
739                                     ;
740 000004E8 46                         inc     esi             ; next page table
741 000004E9 E2EB                       loop    mi_10           ; allocate next kernel page table
742                                                             ; (ecx = page table count + 1)
743                                     ;
744 000004EB 59                         pop     ecx             ; (**) PDE count (= pg. tbl. count)
745                                     ;
746                                     ; Initialize Kernel Page Directory and Kernel Page Tables
747                                     ;
748                                     ; Initialize Kernel's Page Directory
749 000004EC 8B3D[80760100]             mov     edi, [k_page_dir]
750 000004F2 89F8                       mov     eax, edi
751 000004F4 0C03                       or      al, PDE_A_PRESENT + PDE_A_WRITE
752                                                             ; supervisor + read&write + present
753 000004F6 89CA                       mov     edx, ecx        ; (**) PDE count (= pg. tbl. count)
754                                 mi_11:
755 000004F8 0500100000                 add     eax, 4096       ; Add page size (PGSZ)
756                                                              ; EAX points to next page table
757 000004FD AB                         stosd
758 000004FE E2F8                       loop    mi_11
759 00000500 29C0                       sub     eax, eax        ; Empty PDE
760                                     ;mov    cx, 1024        ; Entry count (PGSZ/4)
761                                     ; 29/11/2023
762 00000502 B504                       mov     ch, 4 ; cx = 4*256 = 1024
763 00000504 29D1                       sub     ecx, edx
764 00000506 7402                       jz      short mi_12
765 00000508 F3AB                       rep     stosd           ; clear remain (empty) PDEs
766                                     ;
767                                     ; 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
0000050A 8B0D[84760100]                 mov     ecx, [memory_size] ; memory size in pages
00000510 89CA                           mov     edx, ecx        ; (***)
00000512 B003                           mov     al, PTE_A_PRESENT + PTE_A_WRITE
                                                                ; supervisor + read&write + present
                                    mi_13:
00000514 AB                             stosd
00000515 0500100000                     add     eax, 4096
0000051A E2F8                           loop    mi_13
                                        ;and    dx, 1023        ; (***)
                                        ; 30/08/2023
0000051C 81E2FF030000                   and     edx, 1023
00000522 7408                           jz      short mi_14
                                        ;mov    cx, 1024
                                        ; 30/08/2023
00000524 B504                           mov     ch, 4 ; 4*256 = 1024
                                        ;sub    cx, dx          ; from dx (<= 1023) to 1024
                                        ; 24/07/2022
00000526 29D1                           sub     ecx, edx
00000528 31C0                           xor     eax, eax
0000052A F3AB                           rep     stosd           ; clear remain (empty) PTEs
                                                                ; of the last page table
                                    mi_14:
                                        ;  Initialization of Kernel's Page Tables is OK, here.
                                        ;
0000052C 89F8                           mov     eax, edi        ; end of the last page table page
                                                                 ; (beginging of user space pages)
                                                                ;
0000052E C1E80F                         shr     eax, 15         ; convert to M.A.T. byte offset
00000531 24FC                           and     al, 0FCh        ; clear bit 0 and bit 1 for
                                                                ; aligning on dword boundary
00000533 A3[94760100]                   mov     [first_page], eax
00000538 A3[8C760100]                   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
                                        ;
0000053D A1[80760100]                       mov     eax, [k_page_dir]
00000542 0F22D8                         mov     cr3, eax
00000545 0F20C0                         mov     eax, cr0
00000548 0D00000080                     or      eax, 80000000h ; set paging bit (bit 31)
0000054D 0F22C0                         mov     cr0, eax
                                            ;jmp    KCODE:StartPMP

00000550 EA                             db 0EAh                 ; Opcode for far jump
00000551 [57050000]                         dd StartPMP                 ; 32 bit offset
00000555 0800                           dw KCODE                ; kernel code segment descriptor

                                    StartPMP:
                                        ; 06/11//2014
                                        ; Clear video page 0
                                        ;
                                        ; Temporary Code
                                        ;
00000557 B9E8030000                     mov     ecx, 80*25/2
0000055C BF00800B00                     mov     edi, 0B8000h
                                        ; 30/01/2016
                                        ;xor    eax, eax        ; black background, black fore color
00000561 B800070007                     mov     eax, 07000700h  ; black background, light gray fore color
00000566 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)
00000568 BE[2A390100]                   mov     esi, starting_msg
                                        ;; 14/08/2015 (kernel version message will appear
                                        ;;             when protected mode and paging is enabled)
0000056D 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:
00000572 B40A                           mov     ah, 0Ah ; Black background, light green forecolor
                                    ;pkv_2:
                                        ; 20/08/2014
00000574 E8BE030000                     call    printk

                                        ; 'UNIX v7/x86' source code by Robert Nordier (1999)
                                        ; // Set IRQ offsets
                                        ;
                                        ;  Linux (v0.12) source code by Linus Torvalds (1991)
                                        ;
                                                                        ;; ICW1
00000579 B011                           mov     al, 11h                 ; Initialization sequence
0000057B E620                           out     20h, al                 ;       8259A-1
                                        ; jmp   $+2
0000057D E6A0                           out     0A0h, al                ;       8259A-2
                                                                        ;; ICW2
0000057F B020                           mov     al, 20h                 ; Start of hardware ints (20h)
00000581 E621                           out     21h, al                 ;       for 8259A-1
                                        ; jmp   $+2
00000583 B028                           mov     al, 28h                 ; Start of hardware ints (28h)
00000585 E6A1                           out     0A1h, al                ;       for 8259A-2
                                                                        ;
00000587 B004                           mov     al, 04h                 ;; ICW3
00000589 E621                           out     21h, al                 ;       IRQ2 of 8259A-1 (master)
                                        ; jmp   $+2
0000058B B002                           mov     al, 02h                 ;       is 8259A-2 (slave)
0000058D E6A1                           out     0A1h, al                ;
                                                                        ;; ICW4
0000058F B001                           mov     al, 01h                 ;
00000591 E621                           out     21h, al                 ;       8086 mode, normal EOI
                                        ; jmp   $+2
00000593 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)
00000595 BE[C8350100]               mov     esi, ilist
0000059A 8D3D[98730100]             lea     edi, [idt]
                                    ; 26/03/2015
000005A0 B930000000                 mov     ecx, 48         ; 48 hardware interrupts (INT 0 to INT 2Fh)
                                    ; 02/04/2015
000005A5 BB00000800                 mov     ebx, 80000h
                                rp_sidt1:
000005AA AD                         lodsd
000005AB 89C2                       mov     edx, eax
000005AD 66BA008E                   mov     dx, 8E00h
000005B1 6689C3                     mov     bx, ax
000005B4 89D8                       mov     eax, ebx        ; /* selector = 0x0008 = cs */
                                                            ; /* interrupt gate - dpl=0, present */
000005B6 AB                         stosd   ; selector & offset bits 0-15
000005B7 89D0                       mov     eax, edx
000005B9 AB                         stosd   ; attributes & offset bits 16-23
000005BA E2EE                       loop    rp_sidt1
                                    ; 15/04/2016
                                    ; TRDOS 386 (TRDOS v2.0) /// 32 sofware interrupts ///
                                    ;mov    cl, 16          ; 16 software interrupts (INT 30h to INT 3Fh)
000005BC B120                       mov     cl, 32          ; 32 software interrupts (INT 30h to INT 4Fh)
                                rp_sidt2:
000005BE AD                         lodsd
000005BF 21C0                       and     eax, eax
000005C1 7413                       jz      short rp_sidt3
000005C3 89C2                       mov     edx, eax
000005C5 66BA00EE                   mov     dx, 0EE00h      ; P=1b/DPL=11b/01110b
000005C9 6689C3                     mov     bx, ax
000005CC 89D8                       mov     eax, ebx        ; selector & offset bits 0-15
000005CE AB                         stosd
000005CF 89D0                       mov     eax, edx
000005D1 AB                         stosd
000005D2 E2EA                       loop    rp_sidt2
000005D4 EB16                       jmp     short sidt_OK
                                rp_sidt3:
000005D6 B8[AE0D0000]               mov     eax, ignore_int
000005DB 89C2                       mov     edx, eax
000005DD 66BA00EE                   mov     dx, 0EE00h      ; P=1b/DPL=11b/01110b
000005E1 6689C3                     mov     bx, ax
000005E4 89D8                       mov     eax, ebx        ; selector & offset bits 0-15
                                rp_sidt4:
000005E6 AB                         stosd
000005E7 92                         xchg    eax, edx
000005E8 AB                         stosd
000005E9 92                         xchg    edx, eax
000005EA E2FA                       loop    rp_sidt4
                                sidt_OK:
000005EC 0F011D[EE650000]           lidt    [idtd]
                                    ;
                                    ; TSS descriptor setup ; 24/03/2015
000005F3 B8[18760100]               mov     eax, task_state_segment
000005F8 66A3[9A650000]             mov     [gdt_tss0], ax
000005FE C1C010                     rol     eax, 16
00000601 A2[9C650000]               mov     [gdt_tss1], al
00000606 8825[9F650000]             mov     [gdt_tss2], ah
0000060C 66C705[7E760100]68-        mov     word [tss.IOPB], tss_end - task_state_segment
00000614 00
                                    ;
                                    ; IO Map Base address (When this address points
                                    ; to end of the TSS, CPU does not use IO port
                                    ; permission bit map for RING 3 IO permissions,
                                    ; access to any IO ports in ring 3 will be forbidden.)
                                    ;
                                    ;mov    [tss.esp0], esp ; TSS offset 4
                                    ;mov    word [tss.ss0], KDATA ; TSS offset 8 (SS)
00000615 66B82800                   mov     ax, TSS  ; It is needed when an interrupt
                                                     ; occurs (or a system call -software INT- is requested)
                                                     ; while cpu running in ring 3 (in user mode).
                                                     ; (Kernel stack pointer and segment will be loaded
                                                     ; from offset 4 and 8 of the TSS, by the CPU.)
00000619 0F00D8                     ltr     ax  ; Load task register
                                    ;
                                esp0_set0:

                                ; 29/11/2023 - Erdogan Tan
                                ; ------------------------
                                ; If we read following -disabled- stack page setting code...
                                ; When the memory size >= 3GB, one of page tables conflicts with the stack page
                                ; at 4MB-4096 address. So, to leave stack pointer at 90000h is better/default.
                                ; (Problem may not appears for <= 2.5GB main memory but following code is also
                                ; defective because kernel stack page would be seen as unallocated in the M.A.T.
                                ; without by adding a memory allocation code.)
                                ;
                                ; 1st 4MB layout: 1MB kernel -base- reserved + max. 128 KB M.A.T. (at 100000h)
                                ;                + Kernel's page directory (4KB) -just after the M.A.T.-
                                ;             + Kernel's page tables (main memory size / 1024)
                                ; Note:
                                ; 1 or 2 additional kernel page table(s) may be needed for Linear Frame Buffer
                                ; .. but, it/they will not have to be contiguous with other kernel page tables.
                                ; ------------------------

                                ; 27/11/2023 - TRDOS 386 v2.0.7
                                %if 0
                                    ; 30/07/2015
                                    mov     ecx, [memory_size] ; memory size in pages
                                    shl     ecx, 12 ; convert page count to byte count
                                    cmp     ecx, CORE ; beginning of user's memory space (400000h)
                                                      ; (kernel mode virtual address)
                                    jna     short esp0_set1
                                    ;
                                    ; If available memory > CORE (end of the 1st 4 MB)
                                    ; set stack pointer to CORE
                                    ;(Because, PDE 0 is reserved for kernel space in user's page directory)
                                    ;(PDE 0 points to page table of the 1st 4 MB virtual address space)
                                    mov     ecx, CORE
                                esp0_set1:
                                    mov     esp, ecx ; top of kernel stack (**tss.esp0**)
                                %endif

                                esp0_set_ok:
                                    ; 30/07/2015 (**tss.esp0**)
```

```
1015 0000061C 8925[1C760100]            mov     [tss.esp0], esp         ; 90000h ; 29/11/2023
1016                                                                    ; <-- 97000h ; 04/12/2023 (max. 3072 bytes)
1017 00000622 66C705[20760100]10-           mov     word [tss.ss0], KDATA
1017 0000062A 00
1018                                    ; 14/08/2015
1019                                    ; 10/11/2014 (Retro UNIX 386 v1 - Erdogan Tan)
1020                                    ;
1021                                    ;cli    ; Disable interrupts (for CPU)
1022                                    ;    (CPU will not handle hardware interrupts, except NMI!)
1023                                    ;
1024 0000062B 30C0                      xor     al, al          ; Enable all hardware interrupts!
1025 0000062D E621                      out     21h, al         ; (IBM PC-AT compatibility)
1026 0000062F EB00                      jmp     $+2             ; (All conventional PC-AT hardware
1027 00000631 E6A1                      out     0A1h, al        ;  interrupts will be in use.)
1028                                                            ; (Even if related hardware component
1029                                                            ;  does not exist!)
1030                                    ; Enable NMI
1031 00000633 B07F                      mov     al, 7Fh         ; Clear bit 7 to enable NMI (again)
1032 00000635 E670                      out     70h, al
1033                                    ; 23/02/2015
1034 00000637 90                        nop
1035 00000638 E471                      in      al, 71h         ; read in 71h just after writing out to 70h
1036                                                            ; for preventing unknown state (!?)
1037                                    ;
1038                                    ; Only a NMI can occur here... (Before a 'STI' instruction)
1039                                    ;
1040                                    ; 02/09/2014
1041                                    ;xor    bx, bx
1042                                    ; 24/07/2022
1043 0000063A 31DB                      xor     ebx, ebx
1044 0000063C 66BA0002                  mov     dx, 0200h       ; Row 2, column 0  ; 07/03/2015
1045 00000640 E82E1D0000                call    _set_cpos       ; 24/01/2016
1046
1047                                    ; 14/11/2020 (TRDOS 386 v2.0.3)
1048                                    ; Check VBE3 protected mode interface/feature(s)
1049
1050                                    ;cmp    byte [vbe3], 3 ; 03h
1051                                        ;jne    short display_mem_info
1052
1053                                        ; 20/11/2020
1054 00000645 803D[86090000]02              cmp     byte [vbe3], 2 ; 02h
1055 0000064C 7707                          ja      short vbe3_pmid_chk
1056                                        ;;jb    short display_mem_info
1057                                        ;jb     display_mem_info ; 02/12/2020
1058 0000064E 7220                          jb      short jmp_display_mem_info ; 24/07/2022
1059 00000650 E9EE010000                    jmp     check_boch_plex86_vbe
1060
1061                                    vbe3_pmid_chk:
1062 00000655 B9EA7F0000                    mov     ecx, 32768 - (20+2) ; 32766 - PMInfoBlockSize
1063 0000065A BE02000C00                    mov     esi, 0C0002h ; 1st word of the video bios rom is 0AA55h
1064
1065                                    chk_pmi_sign:
1066                                        ;mov    eax, [esi]
1067                                        ;cmp    eax, 'PMID'
1068                                        ; 30/11/2020
1069                                        ;cmp    al, 'P'
1070                                        ;jne    short chk_pmi_sign_next
1071 0000065F 813E504D4944                  cmp     dword [esi], 'PMID'
1072                                        ;je     short display_vbios_product_name
1073 00000665 740E                          je      short verify_pmib_chksum ; 15/11/2020
1074                                    ;chk_pmi_sign_next:
1075 00000667 46                            inc     esi  ; inc si
1076 00000668 E2F5                          loop    chk_pmi_sign
1077
1078                                    not_valid_pmib:
1079 0000066A FE0D[86090000]                dec     byte [vbe3] ; 2 = VBE2 compatible
1080                                                                ; (vbe3 feature is defective in this vbios)
1081                                        ;jmp    short display_mem_info
1082                                    jmp_display_mem_info:       ; 24/07/2022
1083                                        ; 02/12/2020
1084 00000670 E987010000                    jmp     display_mem_info
1085
1086                                    verify_pmib_chksum:
1087                                        ; 18/10/2023
1088                                        ; (ATI RV370 video bios contains ZERO at Checksum offset.)
1089 00000675 31C0                          xor     eax, eax
1090 00000677 384613                        cmp     byte [esi+19], al ; 0 ; Checksum byte
1091 0000067A 7611                          jna     short skip_verify_pmib_chksum ; may be ATI video bios
1092                                        ;cmp    dword [esi+16], 0C000h
1093                                        ;       ; CodeSegSel = 0C000h, InProtectMode = 0, Checksum = 0
1094                                        ;je     short skip_verify_pmib_chksum ; may be ATI video bios
1095                                        ; 18/10/2023
1096                                        ; 15/11/2020
1097                                        ;xor    eax, eax
1098                                        ;;mov   ecx, eax
1099                                        ;mov    cl, 20
1100 0000067C 66B91400                      mov     cx, 20 ; 30/11/2020
1101 00000680 56                            push    esi
1102                                    pmib_sum_bytes:
1103 00000681 AC                            lodsb
1104 00000682 00C4                          add     ah, al
1105 00000684 E2FB                          loop    pmib_sum_bytes
1106 00000686 5E                            pop     esi
1107 00000687 08E4                          or      ah, ah
1108 00000689 75DF                          jnz     short not_valid_pmib ; AH must be 0
1109
1110                                        ; 18/10/2023
1111 0000068B 30C0                          xor     al, al  ; eax = 0
1112
1113                                        ; 28/02/2021
1114                                        ; Set default (initial) truecolor bpp value to 32
1115                                        ; (for VBE3 video bios.. because vbe3 video bioses
1116                                        ;  use 32bpp -for truecolor modes- instead of 24bpp)
1117                                        ; (This setting may be changed via 'sysvideo' bx=0908h)
1118
1119                                    skip_verify_pmib_chksum: ; 18/10/2023
1120
1121 0000068D C605[43730100]20              mov     byte [truecolor], 32 ; (RGB: 00RRGGBBh)
1122
1123                                    display_vbios_product_name: ; 14/11/2020
1124
1125                                        ; ESI points to 'PMID' (0C0000h + 'PMID' offset)
1126
1127                                        ; 15/11/2020
1128                                        ;mov    [pmid_addr], si         ; PMInfoBlock offset
1129                                        ;               ; (in VGA bios, 0C0000h + offset)
1130                                        ; 02/12/2020
1131                                        ;push   esi ; * pmid_addr
1132 00000694 89F7                          mov     edi, esi
1133
1134                                        ;mov    esi, [VBE3INFOBLOCK+22] ; 097E00h + 16h
1135                                        ;               ; OemVendorNamePtr (seg16:off16)
1136 00000696 8B35067E0900                  mov     esi, [VBE3INFOBLOCK+6] ; 097E00h + 06h
1137                                                        ; OemStringPtr (seg16:off16)
```

```
                                    ; 18/10/2023
                                    ;xor    al, al  ; eax = 0
0000069C 6696                       xchg    ax, si  ; ax = offset, si = 0
0000069E C1EE0C                     shr     esi, 12 ; (to convert segment to base addr)
000006A1 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
000006A4 B40C                       mov     ah, 0Ch ; Black background, light red forecolor

000006A6 E8D63A0000                 call    print_kmsg

                                    ;mov    ah, 07h

000006AB BE[25730100]               mov     esi, vesa_vbe3_bios_msg
                                    ;call   print_kmsg
000006B0 E8D23A0000                 call    pkmsg_loop ; 30/11/2020

                                    ; 02/12/2020
                                    ;pop    edi ; * pmid_addr

                                ; 04/12/2023 - TRDOS 386 v2.0.7
                                %if 0
                                    ; 24/07/2022
                                    ; 29/11/2020
                                vbe3pminit:
                                    ; 30/11/2020
                                    ;cmp    byte [vbe3], 3 ; is VESA VBE3 PMI ready ?
                                    ;jne    short di4

                                    ; Allocate 64KB contiguous (kernel) memory block
                                    xor     eax, eax
                                    mov     ecx, 65536
                                    call    allocate_memory_block
                                    ;jc     short di4
                                    ;jc     di0 ; 30/11/2020
                                    ; 24/07/2022
                                    jnc     short vbe3pminit0
                                    jmp     di0

                                vbe3pminit0:
                                    ; of course this block must be in the 1st 16MB
                                    ; because vbe3 pmi segments will be 16 bit segments
                                    ; (80286 type segment descriptors in GDT)

                                    mov     [vbe3bios_addr], eax
                                %endif

                                    ; 04/12/2023 - TRDOS 386 v2.0.7      ; (+!*!+)
                                    ; fixed PM-VBIOS address (no need to add a memory block)
                                    ; in the reserved -and already allocated- area under 1MB
                                    ; (purpose: to prevent page faults if memory size > 2.5GB)
                                    ; ((User's page dir contains only the 1st 4MB of system mem
                                    ; as PDE. So, this causes to page faults during an interrupt
                                    ; in user mode, because if memory size > 2.5GB, kernel
                                    ; page tables overs/passes 4MB limit and PM-VBIOS
                                    ; is located after kernel page tables.))
                                vbe3pminit:
000006B5 B800000600                 mov     eax, VBE3BIOSCODE_ADDR ; 60000h      ; (+!*!+)

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

                                    ; 30/11/2020
000006BF 29F7                       sub     edi, esi ; izolate offset
000006C1 01C7                       add     edi, eax ; new address
000006C3 893D[F49C0100]             mov     [pmid_addr], edi ; new 'PMID' location

                                    ; Move VIDEO BIOS from 0C0000h to EAX
000006C9 B900400000                 mov     ecx, 65536/4
000006CE 89C7                       mov     edi, eax ; 30/11/2020
000006D0 F3A5                       rep     movsd

                                    ; 02/12/2020
                                    ; 30/11/2020
                                    ; set vbe3 segment selectors

                                ; 04/12/2023 - TRDOS 386 v2.0.7   ; (+!*!+)
                                %if 0
                                    ; VBE3CS (VESA VBE3 video bios code segment)
                                    mov     edi, _vbe3_CS+2 ; base address bits 0..15
                                    stosw   ; edi = _vbe3_CS+4
                                    ror     eax, 16
                                    mov     [edi], al ; base address, bits 16..23

                                    ; VBE3DS ('CodeSegSel' in PMInfoBlock)
                                    mov     edi, _vbe3_DS+4 ; base addr bits 16..23
                                    mov     [edi], al
                                    rol     eax, 16
                                    mov     [edi-2], ax ; base address, bits 0..15
                                %endif
                                    ; VBE3BDS (BIOSDataSel in PMInfoBlock)
000006D2 BF[AA650000]               mov     edi, _vbe3_BDS+2 ; base addr bits 0..15
000006D7 B800700900                 mov     eax, VBE3BIOSDATABLOCK ; 1536 bytes
000006DC 66AB                       stosw   ; edi = _vbe3_BDS+4
000006DE C1E810                     shr     eax, 16
000006E1 8807                       mov     [edi], al ; base address, bits 16..23

                                    ; VBE3SS (1024 bytes)
000006E3 BF[D2650000]               mov     edi, _vbe3_SS+2 ; base addr bits 0..15
000006E8 B800600900                 mov     eax, VBE3STACKADDR ; size = 1024 bytes
000006ED 66AB                       stosw   ; edi = _vbe3_SS+4
000006EF C1E810                     shr     eax, 16
000006F2 8807                       mov     [edi], al ; base address, bits 16..23

                                    ; stack pointer (esp) will be set to 1020
                                    ; (before VBE3 PMI call)

                                    ; VBE3ES (max: 2048 bytes)
000006F4 BF[DA650000]               mov     edi, _vbe3_ES+2 ; base addr bits 0..15
000006F9 B800760900                 mov     eax, VBE3SAVERESTOREBLOCK
000006FE 66AB                       stosw   ; edi = _vbe3_ES+4
00000700 C1E810                     shr     eax, 16
00000703 8807                       mov     [edi], al ; base address, bits 16..23

```

```
                                    ;Note: low word of _VBE3_ES base address will be
                                    ;      set -again- by VBE3 PMI caller routine

                                    ; 09/12/2020
                                    ;; set pmi32 (as VBE3 PMI is ready)
                                    ;inc   byte [pmi32] ; = 1

                                    ; KCODE16 (set PMI far return segment)
00000705 BF[E2650000]               mov    edi, _16bit_CS+2 ; base addr bits 0..15
0000070A B8[92070000]               mov    eax, pminit_return_addr16
0000070F 66AB                       stosw  ; edi = _16bit_CS+4
00000711 C1E810                     shr    eax, 16
00000714 8807                       mov    [edi], al ; base address, bits 16..23

                                    ; 30/11/2020
                                    ; clear mem from VBE3 BIOS data area emu block
                                    ; to end of vbe3 buffers

                                    ; 18/10/2023 - TRDOS v2.0.7
                                    ; (VBE3BIOSDATABLOCK contains a copy of the 1st
                                    ; 1536 bytes of the memory, IVT, ROMBIOS DATA etc.)
                                    ; ((it must not be cleared here))
                                %if 0
                                    ; 01/12/2020
                                    mov    edi, VBE3BIOSDATABLOCK ; 97000h
                                    ;mov   cx, (VBE3INFOBLOCK-VBE3BIOSDATABLOCK)/4
                                    ;      ; ecx = 3584/4 double words
                                    ; 21/12/2020
                                    mov    cx, (VBE3MODEINFOBLOCK-VBE3BIOSDATABLOCK)/4
                                           ; ecx = 3072/4 double words
                                    ;xor   eax, eax
                                    xor    al, al
                                    rep    stosd
                                %endif
                                    ; 18/10/2023 - TRDOS v2.0.7
                                    ; (VBE3BIOSDATABLOCK contains a copy of the 1st
                                    ; 1536 bytes of the memory, IVT, ROMBIOS DATA etc.)
                                    ; ((it must not be cleared here))
00000716 BF00760900                 mov    edi, VBE3SAVERESTOREBLOCK ; 97600h
0000071B 66B98001                   mov    cx, (VBE3MODEINFOBLOCK-VBE3SAVERESTOREBLOCK)/4
                                            ; ecx = 1536/4 = 384 double words
                                    ;xor   eax, eax
0000071F 30C0                       xor    al, al
00000721 F3AB                       rep    stosd

                                    ; Filling PMInfoBlock selector fields
00000723 8B3D[F49C0100]             mov    edi, [pmid_addr]
00000729 66C747083800               mov    word [edi+PMInfo.BIOSDataSel], VBE3BDS
0000072F 66C7470A4000               mov    word [edi+PMInfo.A0000Sel], VBE3A000
00000735 66C7470C4800               mov    word [edi+PMInfo.B0000Sel], VBE3B000
0000073B 66C7470E5000               mov    word [edi+PMInfo.B8000Sel], VBE3B800
00000741 66C747105800               mov    word [edi+PMInfo.CodeSegSel], VBE3DS
00000747 C6471201                   mov    byte [edi+PMInfo.InProtectMode], 1

                                    ; Calculate and write checksum byte
0000074B 89FE                       mov    esi, edi
0000074D B113                       mov    cl, PMInfo.size - 1
                                    ;xor   ah, ah
                                pmid_chksum:
0000074F AC                         lodsb
00000750 00C4                       add    ah, al
00000752 E2FB                       loop   pmid_chksum
00000754 F6DC                       neg    ah ; 1 -> 255, 255 -> 1
00000756 8826                       mov    [esi], ah  ; checksum

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

00000758 668B4706                   mov    ax, [edi+PMInfo.PMInitialize]
                                    ; 30/11/2020
0000075C C1E010                     shl    eax, 16 ; save entry address in hw
                                    ; ax = 0

                                    ; 02/12/2020
0000075F 68[B6070000]               push   pminit_ok ; normal, near return address

                                    ; 30/11/2020
                                _VBE3PMI_fcall:
                                    ; ax = function, hw of eax = entry address
00000764 9C                         pushf  ; save 32 bit flags
00000765 56                         push   esi ; *
00000766 55                         push   ebp ; **

00000767 89E5                       mov    ebp, esp ; save 32 bit stack pointer

                                    ;mov   esi, eax
                                    ; 28/12/2025 (PMI32 BugFix)
00000769 0FB7F0                     movzx  esi, ax

                                    ; Disable interrupts (clear interrupt flag)
                                    ; Reset Interrupt MASK Registers (Master&Slave)
0000076D B0FF                       mov    al, 0FFh       ; mask off all interrupts
0000076F E621                       out    21h, al        ; on master PIC (8259)
00000771 EB00                       jmp    $+2  ; (delay)
00000773 E6A1                       out    0A1h, al       ; on slave PIC (8259)

                                    ; 02/12/2020
00000775 66B86000                   mov    ax, VBE3SS
00000779 8ED0                       mov    ss, ax

0000077B BCFC030000                 mov    esp, 1020 ; 30/11/2020

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

00000780 C1E810                     shr    eax, 16 ; now, entry address is in lw

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

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

                                    ; 30/11/2020 (16 bit stack during retf from video bios)
00000783 C7042400007000             mov    dword [esp], KCODE16 << 16
                                                        ; ip = 0, cs = KCODE16
                                    ; 01/12/2020
                                    ;mov   dword [VBE3STACKADDR+1020], KCODE16*65536

                                    ;mov   [jumpfar16], eax
```

```
                                    ; 02/12/2020
                                    ; 30/11/2020 (32 bit stack during retf from kernel)
                                    ; far jump/call via retf
0000078A 6A30                       push    VBE3CS ; VBE3 video bios's code segment
0000078C 50                         push    eax ; PMInitialize or EntryPoint

                                    ;mov    ax, si ; restore function
                                    ; 24/07/2022
0000078D 89F0                       mov     eax, esi

                                    ; 02/12/2020
0000078F 31F6                       xor     esi, esi ; (not necessary, it is not used)

00000791 CB                         retf    ; far return (to 16 bit code segment)

                                    ; 01/12/2020
                                    ;db     0EAh  ; far jump to 16 bit code segment
                                ;jumpfar16:
                                    ;dd     0
                                    ;dw     VBE3CS

                                ;stack16:
                                    ;dd     1020
                                    ;dw     VBE3SS

                                    align 2

                                pminit_return_addr16:
                                    ; 02/12/2020
                                    ; 30/11/2020
                                    ;;db    66h                 ; Prefix for 32-bit
                                    ;db     0EAh                ; Opcode for far jump
                                    ;dd     pminit_return_addr32 ; 32 bit Offset
                                    ;dw     KCODE               ; 32 bit code segment
                                    ; 01/12/2020
00000792 EA[99070000]0800           jmp     KCODE:pminit_return_addr32

                                pminit_return_addr32:
                                    ; restore 32 bit kernel selectors and 32 bit stack addr
00000799 BE10000000                 mov     esi, KDATA
0000079E 8EDE                       mov     ds, si
000007A0 8EC6                       mov     es, si
000007A2 8ED6                       mov     ss, si
000007A4 89EC                       mov     esp, ebp  ; top of stack = iretd return addr

000007A6 5D                         pop     ebp ; **
000007A7 5E                         pop     esi ; *
000007A8 9D                         popf    ; restore 32 bit flags

                                    ; enable interrupts

000007A9 FA                         cli

                                    ; 21/12/2020
000007AA 50                         push    eax

000007AB 30C0                       xor     al, al     ; Enable all hardware interrupts!
000007AD E621                       out     21h, al    ; (IBM PC-AT compatibility)
000007AF EB00                       jmp     $+2        ; (All conventional PC-AT hardware
000007B1 E6A1                       out     0A1h, al   ;  interrupts will be in use.)
                                                       ; (Even if related hardware component
                                                       ;  does not exist!)
000007B3 58                         pop     eax

000007B4 FB                         sti

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

000007B5 C3                         retn

                                pminit_ok:
                                    ; 03/12/2020
                                    ; (set [pmid_addr] to PMI entry point for next calls)
000007B6 8305[F49C0100]04           add     dword [pmid_addr], PMInfo.EntryPoint ; + 4

                                    ; 17/01/2021
                                    ; copy EDID data from temporary location to final address
000007BD 803D[19420000]4F           cmp     byte [edid], 4Fh
000007C4 7510                       jne     short vbe3h_chcl
                                    ;mov    ecx, 32 ; 128 bytes, 32 dwords
                                    ; 24/07/2022
000007C6 31C9                       xor     ecx, ecx
000007C8 B120                       mov     cl, 32
000007CA BE007D0900                 mov     esi, VBE3EDIDINFOBLOCK ; 97D00h
000007CF BF[6C9C0100]               mov     edi, edid_info
000007D4 F3A5                       rep     movsd
                                    ; 17/01/2021
                                vbe3h_chcl:
                                    ; 16/01/2021
                                    ;; 06/12/2020
                                    ;; Save video mode 03h regs/dac/bios state
                                    ;
                                    ;mov    ax, 4F04h ; VESA VBE Function 04h
                                    ;                 ; Save/Restore State
                                    ;sub    dl, dl  ; 0 = return buffer size
                                    ;mov    cx, 0Fh  ; ctrl/bios/dac/regs
                                    ;
                                    ;call   int10h_32bit_pmi
                                    ;; bx = number of 64-byte blocks to hold the state buff
                                    ;
                                    ;;mov   [vbe3stbufsize], bx
                                    ;; 16/01/2021
                                    ;or     word [vbe3stbsflags], 32768  ; set bit 15
                                    ;mov    ax, bx
                                    ;shl    ax, 6  ; * 64
                                    ;mov    [vbestatebufsize+30], ax
                                    ;
                                    ;; 06/12/2020
                                    ;; check 'vbe3stbufsize' (it must be <= 32)
                                    ;
                                    ;cmp    bx, 32
                                    ;ja     short display_mem_info ; light red forecolor
                                    ;
                                    ;; 16/01/2021
                                    ;or     byte [vbe3stbsflags], 1 ; set bit 0

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

000007D6 BE40810B00                 mov     esi, 0B8000h + 160*2 ; row 2
000007DB 89F7                       mov     edi, esi
```

```
                                vbe3h_chcl_next:
000007DD 66AD                       lodsw
000007DF 80FC0C                     cmp     ah, 0Ch ; light red forecolor
000007E2 7518                       jne     short display_mem_info
000007E4 B40E                       mov     ah, 0Eh ; yellow forecolor
000007E6 66AB                       stosw
000007E8 EBF3                       jmp     short vbe3h_chcl_next

                                di5:
                                    ; 18/10/2023 - TRDOS 386 v2.0.7
                                    ; ATI RV370 Video BIOS (VESA VBE2)
                                    ; has PMI. (as described in VESA VBE3 specification)
                                    ;
000007EA FE05[86090000]             inc     byte [vbe3]  ; [vbe3] = 2 -> 3
                                                         ; will be decreased to 2 again if 'PMID'
                                                         ; signature will not be found.
000007F0 E960FEFFFF                 jmp     vbe3_pmid_chk ; check for ATI Video BIOS

                                di0:
                                    ; 30/11/2020
                                    ; Memory allocation error !
000007F5 C605[86090000]00           mov     byte [vbe3], 0 ; disable VBE3

                                display_mem_info:
                                    ; 19/12/2020
                                    ; temporary
                                    ; 24/11/2023
000007FC 803D[86090000]02           cmp     byte [vbe3], 2
00000803 7205                       jb      short dmi
00000805 E893390000                 call    default_lfb_info
                                dmi:
                                    ; 06/11/2014
0000080A E819390000                 call    memory_info
                                    ; 14/08/2015
                                    ;call   getch ; 28/02/2015

                                    ; 07/12/2023
                                    ; check EDID info for LCD monitor -screen resolution-
                                    ;       for modifying VGA mode 13h CRTC parameters
                                    ;       (if it is needed or not)

0000080F E82D010000                 call    video_mode_13h_parms

                                drv_init:
00000814 FB                         sti     ; Enable Interrupts
                                    ; 06/02/2015
00000815 8B15[7E660000]             mov     edx, [hd0_type] ; hd0, hd1, hd2, hd3
0000081B 668B1D[7C660000]           mov     bx, [fd0_type] ; fd0, fd1
                                    ; 22/02/2015
00000822 6621DB                     and     bx, bx
00000825 756C                       jnz     short di1
                                    ;
00000827 09D2                       or      edx, edx
00000829 757A                       jnz     short di2
                                    ;
                                setup_error:
0000082B BE[CE380100]               mov     esi, setup_error_msg
                                psem:
00000830 AC                         lodsb
00000831 08C0                       or      al, al
                                    ;jz     short haltx ; 22/02/2015
00000833 747C                       jz      short di3
00000835 56                         push    esi
                                    ; 13/05/2016
00000836 BB07000000                 mov     ebx, 7  ; Black background,
                                                    ; light gray forecolor
                                                    ; Video page 0 (BH=0)
0000083B E89B1A0000                 call    _write_tty
00000840 5E                         pop     esi
00000841 EBED                       jmp     short psem

                                check_boch_plex86_vbe:
                                    ; 20/10/2023
                                    ; 18/10/2023
                                    ; 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)

                                    ; 20/10/2023
                                    ; (ATI RV370 video bios has PMI support but
                                    ; it is a VESA VBE2 bios. So, even if [vbe3] is set
                                    ; to 3 for PMI functionality, it is better to display
                                    ; VESA VBE number as it is declared by the video bios.)

00000843 C605[2E730100]32           mov     byte [vbe_vnumber], "2" ; 20/10/2023

0000084A 29C0                       sub     eax, eax ; 0
0000084C 66BACE01                   mov     dx, 1CEh ; VBE_DISPI_IOPORT_INDEX
00000850 66EF                       out     dx, ax ; VBE_DISPI_INDEX_ID register
                                    ;mov    ax, 0B0C0h ; VBE_DISPI_ID0
                                    ;mov    dx, 1CFh ; VBE_DISPI_IOPORT_DATA
00000852 6642                       inc     dx
                                    ;out    dx, ax
                                    ;nop
00000854 66ED                       in      ax, dx
00000856 80FCB0                     cmp     ah, 0B0h
                                    ;jne    short not_boch_qemu_vbe
                                    ; 18/10/2023
                                    ;jne    short display_mem_info
00000859 758F                       jne     short di5  ; ATI VESA VBE2 bios or another

0000085B 3CC5                       cmp     al, 0C5h ; it must be 0B0C4h or 0B0C5h ..
                                    ;ja     short not_boch_qemu_vbe
0000085D 779D                       ja      short display_mem_info
0000085F 3CC0                       cmp     al, 0C0h ; 0B0C0h to 0B0C5h .. ; Qemu
                                    ;cmp    al, 0C4h ; 0BC04h or 0B0C5h is OK ; Bochs
                                    ;jb     short not_boch_qemu_vbe
00000861 7299                       jb      short display_mem_info

                                    ; save VESA VBE2 bios (bochs/qemu) signature
                                    ; for enabling VBE2 functions in TRDOS 386 v2 kernel
00000863 A2[87090000]               mov     [vbe2bios], al ; 0C4h or 0C5h (for BOCHS)
                                                           ; (0C0h-0C5h for QEMU)
                                    ; 20/10/2023
                                    ;mov    byte [vbe_vnumber], "2"

                                    ; 26/11/2020
```

```
                                    ; "BOCHS/QEMU/VIRTUALBOX VBE2 Video BIOS ..".
00000868 BE[10730100]               mov     esi, vbe2_bochs_vbios ; BOCH/QEMU vbios msg
0000086D B40E                       mov     ah, 0Eh  ; Yellow font
0000086F E80D390000                 call    print_kmsg

                                    ; this is not necessary ! (20/11/2020)
00000874 803D[87090000]C4           cmp     byte [vbe2bios], 0C4h
                                    ;jb     display_mem_info      ; (QEMU)
                                    ; 02/12/2023
0000087B 720E                       jb      short not_boch_qemu_vbe

                                    ; Display kernel version message if 0E9h hack port
                                    ; is enabled (bochs emulator feature)
0000087D 66BAE900                   mov     dx, 0E9h ; hack port for BOCHS
00000881 BE[69730100]               mov     esi, kernel_version_msg
                                kvmsg_next_char:
00000886 AC                         lodsb
00000887 08C0                       or      al, al
00000889 7505                       jnz     short put_kvmsg_in_hack_port
                                not_boch_qemu_vbe:
0000088B E96CFFFFFF                 jmp     display_mem_info
                                put_kvmsg_in_hack_port:
00000890 EE                         out     dx, al
00000891 EBF3                       jmp     short kvmsg_next_char

                                di1:
                                    ; supress 'jmp short T6'
                                    ;  (activate fdc motor control code)
00000893 66C705[E8090000]90-        mov     word [T5], 9090h ; nop
0000089B 90
                                    ;
                                    ;mov    ax, int_0Eh    ; IRQ 6 handler
                                    ;mov    di, 0Eh*4      ; IRQ 6 vector
                                    ;stosw
                                    ;mov    ax, cs
                                    ;stosw
                                    ;; 16/02/2015
                                        ;;mov     dword [DISKETTE_INT], fdc_int ; IRQ 6 handler
                                    ;
0000089C E815460000                 CALL    DSKETTE_SETUP  ; Initialize Floppy Disks
                                    ;
000008A1 09D2                       or      edx, edx
000008A3 740C                           jz      short di3
                                di2:
000008A5 E842460000                 call    DISK_SETUP     ; Initialize Fixed Disks
                                        ;jc setup_error
                                    ; 24/07/2022
000008AA 7305                       jnc     short di3
000008AC E97AFFFFFF                 jmp     setup_error
                                di3:
000008B1 E846380000                 call    setup_rtc_int  ; 22/05/2015 (dsectrpm.s)
                                    ;
000008B6 E8A92E0100                 call    display_disks ; 07/03/2015  (Temporary)
                                ;haltx:
                                    ; 14/08/2015
                                    ;call   getch ; 22/02/2015
                                    ;sti    ; Enable interrupts (for CPU)
                                ;   ; 29/01/2016
                                ;   sub     ah, ah ;  read time count
                                ;   call    int1Ah
                                ;   mov     edx, ecx ; 18.2 * seconds
                                ;md_info_msg_wait1:
                                ;   ; 29/01/2016
                                ;   mov     ah, 1
                                ;   call    int16h
                                ;   jz      short md_info_msg_wait2
                                ;   xor     ah, ah ; 0
                                ;       call    int16h
                                ;   jmp     short md_info_msg_ok
                                ;md_info_msg_wait2:
                                ;   sub     ah, ah  ; read time count
                                ;   call    int1Ah
                                ;   cmp     edx, ecx ; ; 18.2 * seconds
                                ;   jna     short md_info_msg_wait3
                                ;   xchg    edx, ecx
                                ;md_info_msg_wait3:
                                ;   sub     ecx, edx
                                ;   cmp     ecx, 127 ; 7 seconds (18.2 * 7)
                                ;   jb      short md_info_msg_wait1
                                ;md_info_msg_ok:

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

000008BB 803D[86090000]02           cmp     byte [vbe3], 2
000008C2 7225                       jb      short di4

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

000008C4 BEFE7B0900                 mov     esi, VBE3MODEINFOBLOCK - 2
000008C9 66C7061801                 mov     word [esi], 0118h ; default vbe mode
                                                              ; 1024*768, 24bpp
000008CE E8D4300000                 call    set_lfbinfo_table

                                    ;;;
                                    ; 28/11/2023
                                    ; 20/10/2023 - TRDOS 386 v2.0.7
000008D3 8B35[FC9C0100]             mov     esi, [LFB_ADDR] ; LFB base address (in bytes)
                                    ;mov    edx, [LFB_SIZE] ; 28/11/2023
                                    ;add    edx, 4095
                                    ;;;
                                    ;mov    edx, ((1024*768*4)+4095)>>12 ; 28/11/2023
000008D9 BAE9070000                 mov     edx, ((1920*1080*4)+4095)>>12 ; 28/11/2023
                                    ; edx = LFB size in pages

                                    ; 29/11/2023
                                    ;mov    ebx, esi
                                    ;;add   ebx, 4095 ; LFB start addr is always in page boundary
                                    ;shr    ebx, 12 ; convert byte address to page address
000008DE 8B0D[84760100]             mov     ecx, [memory_size]
                                    ;cmp    ebx, ecx
                                    ;jnb    short di4 ; LFB addr >= main memory size

                                    ; (set the overlapped pages as allocated for kernel)
                                    ;;;
                                    ; 28/11/2023
                                    ; (and set all of LFB pages in the kernel's page tables)

000008E4 E896590000                 call    allocate_lfb_pages_for_kernel
                                di4:
```

```
                                    ; 08/09/2016
000008E9 0F20C0                     mov     eax, cr0
000008EC A810                       test    al, 10h  ; Bit 4, ET (Extension Type)
000008EE 7408                       jz      short sysinit
                                    ; 27/02/2017
000008F0 FE05[40830100]             inc     byte [fpready]
                                    ; 80387 (FPU) is ready
000008F6 DBE3                       fninit ; Initialize Floating-Point Unit
                                sysinit:
                                    ; 30/06/2015
000008F8 E88D630000                 call    sys_init
                                    ;
                                    ;jmp    cpu_reset ; 22/02/2015
                                hang:
                                    ; 23/02/2015
                                    ;sti                    ; Enable interrupts
000008FD F4                         hlt
                                    ;
                                    ;nop
                                    ;; 03/12/2014
                                    ;; 28/08/2014
                                    ;mov    ah, 11h
                                    ;call   getc
                                    ;jz      _c8
                                    ;
                                    ; 23/02/2015
                                    ; 06/02/2015
                                    ; 07/09/2014
000008FE 31DB                       xor     ebx, ebx
00000900 8A1D[AE760100]             mov     bl, [ptty]      ; active_page
00000906 89DE                       mov     esi, ebx
                                    ;shl    si, 1
                                    ; 24/07/2022
00000908 D1E6                       shl     esi, 1
0000090A 81C6[B0760100]             add     esi, ttychr
00000910 668B06                     mov     ax, [esi]
00000913 6621C0                     and     ax, ax
                                    ;jz     short _c8
00000916 74E5                       jz      short hang
00000918 66C7060000                 mov     word [esi], 0
0000091D 80FB03                     cmp     bl, 3           ; Video page 3
                                    ;jb     short _c8
00000920 72DB                       jb      short hang
                                    ;
                                    ; 13/05/2016
                                    ; 07/09/2014
                                nxtl:
                                    ;push   bx
                                    ; 18/04/2021
00000922 53                         push    ebx
00000923 66BB0E00                   mov     bx, 0Eh         ; Yellow character
                                                            ; on black background
                                                            ; bh = 0 (video page 0)
                                                            ; Retro UNIX 386 v1 - Video Mode 0
                                                            ; (PC/AT Video Mode 3 - 80x25 Alpha.)
                                    ;push   ax
                                    ; 18/04/2021
00000927 50                         push    eax
00000928 E8AE190000                 call    _write_tty
                                    ;pop    ax
                                    ; 18/04/2021
0000092D 58                         pop     eax
                                    ;pop    bx
0000092E 5B                         pop     ebx
0000092F 3C0D                       cmp     al, 0Dh         ; carriage return (enter)
                                    ;jne    short _c8
00000931 75CA                       jne     short hang
00000933 B00A                       mov     al, 0Ah         ; next line
00000935 EBEB                       jmp     short nxtl

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

                                    ; 27/08/2014
                                    ; 20/08/2014
                                printk:
                                        ;mov    edi, [scr_row]
                                pkl:
                                    lodsb
00000937 AC                         or      al, al
00000938 08C0                       jz      short pkr
0000093A 7404                       stosw
0000093C 66AB                       jmp     short pkl
0000093E EBF7                   pkr:
                                    retn
00000940 C3

                                ; ************************************
                                video_mode_13h_parms:
                                    ; 07/12/2023 - TRDOS 386 v2.0.7
                                    ; Check EDID information for LCD monitors
                                    ; and change mode 13h parameters if it is required
                                    ; (if resolution > 1280x1024, it is LCD/panel monitor
                                    ; and Video Mode 13h parameters will be modified for 60HZ
                                    ; because LCD monitors does/can not display 320x200 70HZ
                                    ; standard VGA mode)

00000941 803D[19420000]4F           cmp     byte [edid], 4Fh
00000948 753B                       jne     short CRT_monitor

0000094A BE[6C9C0100]               mov     esi, edid_info
0000094F 83C626                     add     esi, 26h ; EDID Standard Timing Identification
00000952 B908000000                 mov     ecx, 8
                                chk_edid:
00000957 66AD                       lodsw
00000959 3C81                       cmp     al, 129 ; (1280/8)-31
0000095B 770E                       ja      short LCD_monitor ; 16:9
0000095D 3C01                       cmp     al, 1
0000095F 7624                       jna     short CRT_monitor
00000961 E2F4                       loop    chk_edid
                                    ;jmp    short CRT_monitor
00000963 C3                         retn
                                mode13h_crtc_60hz:
00000964 0B3EB9858FB8E2             db      0Bh, 3Eh, 0B9h, 85h, 8Fh, 0B8h, 0E2h
                                LCD_monitor:
                                    ; modify default (CRTC register) parameters for 60HZ
                                    ; (320x200 letterbox type screen will appear on LCD screen)
                                    ;
                                    ; Note: When/While [PMI32] -protected mode video bios-
```

```
                                    ;       feature is enabled -by user- for MODE 13h, internal
                                    ;       (TRDOS 386) VGA -video mode setting- parameters
                                    ;       will not be used (will be bypassed)
                                    ;       by TRDOS 386 kernel for all standard VGA modes.
                                    ;       ((NVIDIA/ATI Video Bios PMI will be used.))
                                    ;
                                    ;       ref: github, juj/60hz.cpp
                                    ;            https://gist.github.com/juj/
                                    ;
0000096B B107                       mov     cl, 7
0000096D BE[64090000]               mov     esi, mode13h_crtc_60hz
00000972 BF[BA690000]               mov     edi, vga_mode_13h+10+6 ; CRTC Registers (index: 6)
00000977 A4                         movsb           ; Vertical Total Register
00000978 A4                         movsb           ; Overflow Register
00000979 BF[C4690000]               mov     edi, vga_mode_13h+10+16 ; CRTC Regs (index: 16)
0000097E A4                         movsb           ; Vertical Retrace Start Register
0000097F A4                         movsb           ; Vertical Retrace End Register
00000980 A4                         movsb           ; Vertical Display Enable/End Register
00000981 47                         inc     edi
00000982 47                         inc     edi
00000983 A4                         movsb           ; Vertical Blanking Start Register
00000984 A4                         movsb           ; Vertical Blanking End Register
                                CRT_monitor:
00000985 C3                         retn
                                ; ***********************************

                                ; 14/11/2020 (TRDOS 386 v2.0.3)
00000986 00                     vbe3:       db 0  ; VESA VBE version (must be 03h)
                                          ; for using video bios calls in protected mode
                                vbe2bios:
00000987 B0                         db 0B0h ;
                                ;pmid_addr:
                                    ;dw 0  ; > 0 if 'PMID' sign is found
                                    ;     ;('pmid' offset addr in VGA bios seg, 0C000h)
                                    ;; 02/12/2020
                                    ;dw 0  ; 32 bit address in pmid_addr

                                ; 28/02/2017
                                ; 22/01/2017
                                ; 15/01/2017
                                ; 14/01/2017
                                ; 02/01/2017
                                ; 25/12/2016
                                ; 19/12/2016
                                ; 10/12/2016 (callback)
                                ; 06/06/2016
                                ; 23/05/2016
                                ; 22/05/2016 - TRDOS 386 (TRDOS v2.0) Timer Event Modifications
                                ; 25/07/2015
                                ; 14/05/2015 (multi tasking -time sharing- 'clock', x_timer)
                                ; 17/02/2015
                                ; 06/02/2015 (unix386.s)
                                ; 11/12/2014 - 22/12/2014 (dsectrm2.s)
                                ;
                                ; IBM PC-XT Model 286 Source Code - BIOS2.ASM (06/10/85)
                                ;
                                ;-- HARDWARE INT  08 H - ( IRQ LEVEL 0 ) --------------------------------------
                                ;   THIS ROUTINE HANDLES THE TIMER INTERRUPT FROM FROM CHANNEL 0 OF        :
                                ;   THE 8254 TIMER.  INPUT FREQUENCY IS 1.19318 MHZ AND THE DIVISOR        :
                                ;   IS 65536, RESULTING IN APPROXIMATELY 18.2 INTERRUPTS EVERY SECOND.     :
                                ;                                                                         :
                                ;   THE INTERRUPT HANDLER MAINTAINS A COUNT (40:6C) OF INTERRUPTS SINCE    :
                                ;   POWER ON TIME, WHICH MAY BE USED TO ESTABLISH TIME OF DAY.        :
                                ;   THE INTERRUPT HANDLER ALSO DECREMENTS THE MOTOR CONTROL COUNT (40:40)  :
                                ;   OF THE DISKETTE, AND WHEN IT EXPIRES, WILL TURN OFF THE           :
                                ;   DISKETTE MOTOR(S), AND RESET THE MOTOR RUNNING FLAGS.                   :
                                ;   THE INTERRUPT HANDLER WILL ALSO INVOKE A USER ROUTINE THROUGH           :
                                ;   INTERRUPT 1CH AT EVERY TIME TICK.  THE USER MUST CODE A           :
                                ;   ROUTINE AND PLACE THE CORRECT ADDRESS IN THE VECTOR TABLE.        :
                                ;-------------------------------------------------------------------------------
                                ;

                                timer_int:  ; IRQ 0
                                ;int_08h:   ; Timer
                                    ; 14/10/2015
                                    ; Here, we are simulating system call entry (for task switch)
                                    ; (If multitasking is enabled,
                                    ; 'clock' procedure may jump to 'sysrelease')

00000988 1E                         push    ds
00000989 06                         push    es
0000098A 0FA0                       push    fs
0000098C 0FA8                       push    gs

0000098E 60                         pushad  ; eax, ecx, edx, ebx, esp -before pushad-, ebp, esi, edi
0000098F 66B91000                   mov     cx, KDATA
                                        mov     ds, cx
00000993 8ED9                           mov     es, cx
00000995 8EC1                           mov     fs, cx
00000997 8EE1                           mov     gs, cx
00000999 8EE9

                                    mov     ecx, cr3
0000099B 0F20D9                     mov     [cr3reg], ecx ; save current cr3 register value/content
0000099E 890D[448F0100]

                                    ; 14/01/2017
000009A4 3B0D[80760100]             cmp     ecx, [k_page_dir]
000009AA 7409                       je      short T3

000009AC 8B0D[80760100]             mov     ecx, [k_page_dir]
000009B2 0F22D9                     mov     cr3, ecx
                                T3:
                                    ;sti                            ; INTERRUPTS BACK ON
000009B5 66FF05[00770100]           INC     word [TIMER_LOW]        ; INCREMENT TIME
000009BC 7507                       JNZ     short T4                ; GO TO TEST_DAY
000009BE 66FF05[02770100]           INC     word [TIMER_HIGH]       ; INCREMENT HIGH WORD OF TIME
                                T4:                                 ; TEST_DAY
000009C5 66833D[02770100]18         CMP     word [TIMER_HIGH],018H; TEST FOR COUNT EQUALING 24 HOURS
000009CD 7519                       JNZ     short T5                ; GO TO DISKETTE_CTL
000009CF 66813D[00770100]B0-        CMP     word [TIMER_LOW],0B0H
000009D7 00
000009D8 750E                       JNZ     short T5                ; GO TO DISKETTE_CTL

                                ;-----      TIMER HAS GONE 24 HOURS
                                    ;;SUB   AX,AX
                                    ;MOV    [TIMER_HIGH],AX
                                    ;MOV    [TIMER_LOW],AX
000009DA 29C0                       sub     eax, eax
000009DC A3[00770100]               mov     [TIMER_LH], eax
                                    ;
000009E1 C605[04770100]01           MOV     byte [TIMER_OFL],1

                                ;-----      TEST FOR DISKETTE TIME OUT

                                T5:
```

```
                                    ; 23/12/2014
000009E8 EB1D                       jmp     short T6               ; will be replaced with nop, nop
                                                                   ; (9090h) if a floppy disk
                                                                   ; is detected.
                                    ;mov    al,[CS:MOTOR_COUNT]
000009EA A0[07770100]               mov     al, [MOTOR_COUNT]
000009EF FEC8                       dec     al
                                    ;mov    [CS:MOTOR_COUNT], al  ; DECREMENT DISKETTE MOTOR CONTROL
000009F1 A2[07770100]               mov     [MOTOR_COUNT], al
                                    ;mov    [ORG_MOTOR_COUNT], al
000009F6 750F                       JNZ     short T6               ; RETURN IF COUNT NOT OUT
000009F8 B0F0                       mov     al,0F0h
                                    ;AND    [CS:MOTOR_STATUS],al  ; TURN OFF MOTOR RUNNING BITS
000009FA 2005[06770100]             and     [MOTOR_STATUS], al
                                    ;and    [ORG_MOTOR_STATUS], al
00000A00 B00C                       MOV     AL,0CH                 ; bit 3 = enable IRQ & DMA,
                                                                   ; bit 2 = enable controller
                                                                   ;       1 = normal operation
                                                                   ;       0 = reset
                                                                   ; bit 0, 1 = drive select
                                                                   ; bit 4-7 = motor running bits
00000A02 66BAF203                   MOV     DX,03F2H               ; FDC CTL PORT
00000A06 EE                         OUT     DX,AL                  ; TURN OFF THE MOTOR
                                T6:
                                    ;inc    word [CS:wait_count]   ; 22/12/2014 (byte -> word)
                                                                   ; TIMER TICK INTERRUPT
                                    ;;inc   word [wait_count] ;;27/02/2015
                                    ;INT    1CH                    ; TRANSFER CONTROL TO A USER ROUTINE
                                    ;cli
00000A07 E85E040000                 call    u_timer                ; TRANSFER CONTROL TO A USER ROUTINE
                                    ; 23/05/2016
00000A0C E804110100                 call    clock                  ; Multi Tasking control procedure
                                T7:
                                    ; 14/10/2015
00000A11 B020                       MOV     AL,EOI                 ; GET END OF INTERRUPT MASK
00000A13 FA                         CLI                            ; DISABLE INTERRUPTS TILL STACK CLEARED
00000A14 E620                       OUT     INTA00,AL              ; END OF INTERRUPT TO 8259 - 1
                                    ;
                                    ;;;
                                    ; 21/08/2024
00000A16 803D[11830100]FF           cmp     byte [p_change], 0FFh ; CTRL+BREAK signature
00000A1D 7424                       je      short T9 ; current program will be forced to sysrelease
                                    ;;;
                                rtc_int_2:
                                    ; 26/12/2016
                                    ;mov    ecx, [cr3reg]
                                    ; 13/01/2017
00000A1F 803D[AC8E0100]00           cmp     byte [u.t_lock], 0    ; T_LOCK
00000A26 7730                       ja      short timer_int_return  ; Timer Lock : 'sysrele' is needed !
                                    ; 28/02/2017
                                    ; We need to exit if the user's IRQ callback service is in progress!
                                    ; (To prevent a conflict!)
00000A28 803D[B08E0100]00           cmp     byte [u.r_lock], 0    ; R_LOCK, IRQ callback service lock !
00000A2F 7727                       ja      short timer_int_return  ; Timer Lock : 'sysrele' is needed !
                                    ; 15/01/2017
00000A31 803D[10830100]02           cmp     byte [priority], 2
00000A38 733A                       jnb     short T8  ; current process has a timer event (15/01/2017)
                                    ; 22/05/2016
00000A3A 803D[11830100]00           cmp     byte [p_change], 0 ; in 'set_run_sequence', in 'rtc_p'
00000A41 7615                       jna     short timer_int_return ; 23/05/2016

                                    ; 15/01/2017
                                    ; present process must be changed with high priority process
                                T9:         ; 21/08/2024
                                    ;xor    al, al
00000A43 31C0                       xor     eax, eax ; 26/12/2016
00000A45 A2[11830100]               mov     [p_change], al ; 0
                                    ;mov    byte [priority], 2 ; 15/01/2017 (there is a timer event)

00000A4A 803D[288E0100]FF           cmp     byte [sysflg], 0FFh ; user or system space ?
00000A51 7416                       je      short rtc_int_3     ; user space ([sysflg]= 0FFh)

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

00000A53 A2[7C8E0100]               mov     [u.quant], al ; 0

                                timer_int_return: ; 23/05/2016 - jump from 'rtc_int' ('rtc_int_2')
00000A58 8B0D[448F0100]             mov     ecx, [cr3reg]  ; previous value/content of cr3 register
00000A5E 0F22D9                     mov     cr3, ecx       ; restore cr3 register content
                                    ;
00000A61 61                         popad ; edi, esi, ebp, temp (icrement esp by 4), ebx, edx, ecx, eax
                                    ;
00000A62 0FA9                       pop     gs
00000A64 0FA1                       pop     fs
00000A66 07                         pop     es
00000A67 1F                         pop     ds
                                    ;
00000A68 CF                         iretd   ; return from interrupt

                                rtc_int_3:
00000A69 FE05[288E0100]             inc     byte [sysflg]  ; now, we are in system space
                                    ;
00000A6F E92BC00000                     jmp     sysrelease ; change running process immediatelly

                                T8:
                                    ; 13/01/2017 (eax -> ebx)
                                    ; callback checking... (19/12/2016)
00000A74 31DB                       xor     ebx, ebx
00000A76 871D[A88E0100]             xchg    ebx, [u.tcb] ; callback address (0 = normal return)
00000A7C 09DB                       or      ebx, ebx
00000A7E 74D8                       jz      short timer_int_return

                                    ; Set user's callback routine as return address from this interrupt
                                    ; and set normal return address as return address from callback
                                    ; routine!!! (19/12/2016)

                                    ; 14/01/2017
                                    ; 13/01/2017 - Timer Lock (T_LOCK)
00000A80 FE05[AC8E0100]             inc     byte [u.t_lock]
00000A86 8A0D[288E0100]             mov     cl, [sysflg]
00000A8C 880D[AD8E0100]             mov     [u.t_mode], cl

00000A92 8B2D[1C760100]             mov     ebp, [tss.esp0] ; kernel stack address (for ring 0)
00000A98 83ED14                     sub     ebp, 20         ; eip, cs, eflags, esp, ss
00000A9B 892D[2C8E0100]             mov     [u.sp], ebp
00000AA1 8925[308E0100]             mov     [u.usp], esp

                                    ;or     word [ebp+8], 200h ; 22/01/2017, force enabling interrupts

00000AA7 8B44241C                   mov     eax, [esp+28] ; pushed eax
00000AAB A3[348E0100]               mov     [u.r0], eax
```

```
00000AB0 E8F1FE0000                call   wswap ; save user's registers & status

                                   ; software int is in ring 0 but timer int must return to ring 3
                                   ; so, ring 3 return address and stack registers
                                   ; (eip, cs, eflags, esp, ss)
                                   ; must be copied to timer int return
                                   ; eip will be replaced by callback service routine address

00000AB5 C605[288E0100]FF          mov    byte [sysflg], 0FFh ; user mode

                                   ; system mode (system call)
                                   ;mov   ebp, [u.sp] ; EIP (u), CS (UCODE), EFLAGS (u),
                                                      ; ESP (u), SS (UDATA)

00000ABC 8B4510                    mov    eax, [ebp+16]  ; SS (UDATA
00000ABF 89E6                      mov    esi, esp
00000AC1 50                        push   eax
00000AC2 50                        push   eax
00000AC3 89E7                      mov    edi, esp
00000AC5 893D[308E0100]            mov    [u.usp], edi
00000ACB B908000000                mov    ecx, ((ESPACE/4) - 4) ; except DS, ES, FS, GS
00000AD0 F3A5                      rep    movsd
00000AD2 B104                      mov    cl, 4
00000AD4 F3AB                      rep    stosd
00000AD6 893D[2C8E0100]            mov    [u.sp], edi
00000ADC 89EE                      mov    esi, ebp
00000ADE B105                      mov    cl, 5 ; EIP (u), CS (UCODE), EFLAGS (u), ESP (u), SS (UDATA)
00000AE0 F3A5                      rep    movsd

00000AE2 8B0D[8C8E0100]            mov    ecx, [u.pgdir]
00000AE8 890D[448F0100]            mov    [cr3reg], ecx

                                   ; 13/01/207 (eax -> ebx)
                                   ; EBX = callback routine address (virtual, not physical address!)

                                   ; 09/01/2017
                                   ; !!! CALLBACK ROUTINE MUST BE ENDED/RETURNED WITH 'sysrele'
                                   ;     system call !!!
                                   ; 25/12/2016
                                   ; Callback Note: (19/12/2016)
                                   ; !!! CALLBACK ROUTINE MUST BE ENDED/RETURNED WITH 'RETN' !!!
                                   ;      pushf ; save flags
                                   ;      <callback service code>
                                   ;      popf  ; restore flags
                                   ;      retn ; return to normal running address
                                   ;

                                   ; 15/01/2017
                                   ; 14/01/2017
                                   ; 13/01/2017 (eax -> ebx)
                                   ; 10/01/2017
                                set_callback_addr:
                                   ; 09/01/2017 (**)
                                   ; 02/01/2017 (*)
                                   ; 25/12/2016 (*)
                                   ; 19/12/2016 (TRDOS 386 feature only!)
                                   ;
                                   ; This routine sets return address
                                   ; to start of user's interrupt
                                   ; service (callback) address
                                   ;; and sets callback 'retn' address to normal
                                   ;; return address of user's running code!
                                   ;
                                   ; INPUT:
                                   ;      EBX = callback routine/service address
                                   ;            (virtual, not physical address!)
                                   ;      [u.sp] = kernel stack, points to
                                   ;               user's EIP,CS,EFLAGS,ESP,SS
                                   ;               registers.
                                   ; OUTPUT:
                                   ;      EIP (user) = callback (service) address
                                   ;      CS (user) = UCODE
                                   ;      EFLAGS (user) = flags before callback
                                   ;       ESP (user) = ESP-4 (user, before callback)
                                   ;      [ESP](user) = EIP (user) before callback
                                   ;
                                   ; Note: If CPU was in user mode while entering
                                   ;      the timer interrupt service routine,
                                   ;      'IRET' will get return to callback routine
                                   ;      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
00000AEE 8B2D[2C8E0100]            mov    ebp, [u.sp]; kernel's stack, points to EIP (user)
00000AF4 895D00                    mov    [ebp], ebx
00000AF7 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
00000AFC 1E                         push    ds ; ** ; 23/05/2016
00000AFD 50                         push    eax ; *
00000AFE 66B81000                   mov     ax, KDATA
00000B02 8ED8                       mov     ds, ax
                                    ;
00000B04 8A25[FE760100]             mov     ah, [RTC_2Hz] ;  2 Hz interrupt to 1 Hz function
00000B0A 80F401                     xor     ah, 1
00000B0D 8825[FE760100]             mov     [RTC_2Hz], ah ; 1 = 0.5 second, 0 = 1 second
00000B13 753B                       jnz     short rtc_int_return ; half second
                                    ; 1 second
                                rtc_int_0:
                                    ; 22/05/2016
00000B15 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
00000B16 06                         push    es
00000B17 0FA0                       push    fs
00000B19 0FA8                       push    gs
00000B1B 60                         pushad  ; eax, ecx, edx, ebx, esp -before pushad-, ebp, esi, edi
00000B1C 66B91000                   mov     cx, KDATA
                                        ;mov    ds, cx ; 06/06/2016
00000B20 8EC1                           mov     es, cx
00000B22 8EE1                           mov     fs, cx
00000B24 8EE9                           mov     gs, cx
                                    ;
00000B26 0F20D9                     mov     ecx, cr3
00000B29 890D[448F0100]             mov     [cr3reg], ecx ; save current cr3 register value/content
                                    ;
00000B2F 803D[AC8E0100]00           cmp     byte [u.t_lock], 0 ; timer lock (callback) status ?
00000B36 7711                       ja      short rtc_int_1          ; yes

                                    ; 15/01/2017
00000B38 3B0D[80760100]             cmp     ecx, [k_page_dir]
00000B3E 7409                       je      short rtc_int_1

00000B40 8B0D[80760100]             mov     ecx, [k_page_dir]
00000B46 0F22D9                     mov     cr3, ecx
                                rtc_int_1:
                                    ; Timer event (kernel) functions must be performed with
                                    ; 1 second intervals - TRDOS 386 (TRDOS v2.0) feature ! -
                                    ;
                                    ; 25/08/2014
00000B49 E818030000                 call    rtc_p  ; 19/05/2016 - major modification

                                    ; 23/05/2016
00000B4E 28E4                       sub     ah, ah ; 0
                                    ; 22/05/2016 - TRDOS 386 timer event modifications
                                rtc_int_return: ; 19/05/2016
                                    ; 22/02/2015 - dsectpm.s
                                    ; [ source: http://wiki.osdev.org/RTC ]
                                    ; read status register C to complete procedure
                                    ;(it is needed to get a next IRQ 8)
00000B50 B00C                       mov     al, 0Ch ;
00000B52 E670                       out     70h, al ; select register C
00000B54 90                         nop
00000B55 E471                       in      al, 71h ; just throw away contents
                                    ; 22/02/2015
00000B57 B020                       MOV     AL,EOI          ; END OF INTERRUPT
                                    ;CLI                    ; DISABLE INTERRUPTS TILL STACK CLEARED
00000B59 E6A0                       OUT     INTB00,AL       ; FOR CONTROLLER #2

                                    ; 23/05/2016
00000B5B B020                       MOV     AL,EOI          ; GET END OF INTERRUPT MASK
00000B5D FA                         CLI                     ; DISABLE INTERRUPTS TILL STACK CLEARED
00000B5E E620                       OUT     INTA00,AL       ; END OF INTERRUPT TO 8259 - 1
                                    ;
                                    ; 23/05/2016
00000B60 20E4                       and     ah, ah
                                    ;jz     rtc_int_2
                                    ; 24/07/2022
00000B62 7505                       jnz     short rtc_int_4
00000B64 E9B6FEFFFF                 jmp     rtc_int_2
                                rtc_int_4:
                                    ; ah = 1 (half second)
00000B69 58                         pop     eax ; *
00000B6A 1F                         pop     ds  ; **
00000B6B CF                         iretd
                                
                                ; ////////////////
                                
                                    ; 28/08/2014
                                irq0:
                                        push       dword 0
                                    jmp     short which_irq
                                irq1:
                                        push       dword 1
                                    jmp     short which_irq
                                irq2:
                                        push       dword 2
                                    jmp     short which_irq
                                irq3:
                                    ; 20/11/2015
                                    ; 24/10/2015
00000B78 2EFF15[9F1C0100]           call    dword [cs:com2_irq3]
00000B7F 6A03                       push    dword 3
00000B81 EB35                       jmp     short which_irq
                                irq4:
                                    ; 20/11/2015
                                    ; 24/10/2015
00000B83 2EFF15[9B1C0100]           call    dword [cs:com1_irq4]
00000B8A 6A04                           push       dword 4
00000B8C EB2A                       jmp     short which_irq
                                irq5:
00000B8E 6A05                           push       dword 5
00000B90 EB26                       jmp     short which_irq
                                irq6:
00000B92 6A06                           push       dword 6
00000B94 EB22                       jmp     short which_irq
                                irq7:
00000B96 6A07                           push       dword 7
00000B98 EB1E                       jmp     short which_irq
                                irq8:
```

```
2376 00000B9A 6A08                      push       dword 8
2377 00000B9C EB1A                  jmp    short which_irq
2378                                irq9:
2379 00000B9E 6A09                      push       dword 9
2380 00000BA0 EB16                  jmp    short which_irq
2381                                irq10:
2382 00000BA2 6A0A                      push       dword 10
2383 00000BA4 EB12                  jmp    short which_irq
2384                                irq11:
2385 00000BA6 6A0B                      push       dword 11
2386 00000BA8 EB0E                  jmp    short which_irq
2387                                irq12:
2388 00000BAA 6A0C                      push       dword 12
2389 00000BAC EB0A                  jmp    short which_irq
2390                                irq13:
2391 00000BAE 6A0D                      push       dword 13
2392 00000BB0 EB06                  jmp    short which_irq
2393                                irq14:
2394 00000BB2 6A0E                      push       dword 14
2395 00000BB4 EB02                  jmp    short which_irq
2396                                irq15:
2397 00000BB6 6A0F                      push       dword 15
2398                                ;jmp   short which_irq
2399
2400                                    ; 22/01/2017
2401                                    ; 19/10/2015
2402                                    ; 29/08/2014
2403                                    ; 21/08/2014
2404                                which_irq:
2405 00000BB8 870424                    xchg   eax, [esp]  ; 28/08/2014
2406 00000BBB 53                        push   ebx
2407 00000BBC 56                        push   esi
2408 00000BBD 57                        push   edi
2409 00000BBE 1E                        push   ds
2410 00000BBF 06                        push   es
2411                                    ;
2412 00000BC0 88C3                      mov    bl, al
2413                                    ;
2414 00000BC2 B810000000                mov    eax, KDATA
2415 00000BC7 8ED8                      mov    ds, ax
2416 00000BC9 8EC0                      mov    es, ax
2417                                    ; 19/10/2015
2418 00000BCB FC                        cld
2419                                        ; 27/08/2014
2420 00000BCC 8105[C0350100]A000-           add     dword [scr_row], 0A0h
2420 00000BD4 0000
2421                                    ;
2422 00000BD6 B417                      mov    ah, 17h ; blue (1) background,
2423                                                   ; light gray (7) forecolor
2424 00000BD8 8B3D[C0350100]                mov     edi, [scr_row]
2425 00000BDE B049                      mov    al, 'I'
2426 00000BE0 66AB                      stosw
2427 00000BE2 B052                      mov    al, 'R'
2428 00000BE4 66AB                      stosw
2429 00000BE6 B051                      mov    al, 'Q'
2430 00000BE8 66AB                      stosw
2431 00000BEA B020                      mov    al, ' '
2432 00000BEC 66AB                      stosw
2433 00000BEE 88D8                      mov    al, bl
2434 00000BF0 3C0A                      cmp    al, 10
2435 00000BF2 7208                      jb     short ii1
2436 00000BF4 B031                      mov    al, '1'
2437 00000BF6 66AB                      stosw
2438 00000BF8 88D8                      mov    al, bl
2439 00000BFA 2C0A                      sub    al, 10
2440                                ii1:
2441 00000BFC 0430                      add    al, '0'
2442 00000BFE 66AB                      stosw
2443 00000C00 B020                      mov    al, ' '
2444 00000C02 66AB                      stosw
2445 00000C04 B021                      mov    al, '!'
2446 00000C06 66AB                      stosw
2447 00000C08 B020                      mov    al, ' '
2448 00000C0A 66AB                      stosw
2449                                    ; 23/02/2015
2450 00000C0C 80FB07                    cmp    bl, 7 ; check for IRQ 8 to IRQ 15
2451 00000C0F 7604                      jna    ii2
2452                                    ; 22/01/2017
2453 00000C11 B020                      mov    al, 20h  ; END OF INTERRUPT COMMAND TO
2454 00000C13 E6A0                      out    0A0h, al ; the 2nd 8259
2455                                ii2:
2456 00000C15 B020                      mov    al, 20h  ; END OF INTERRUPT COMMAND TO
2457 00000C17 E620                      out    20h, al ; the 2nd 8259
2458 00000C19 E9CA010000                jmp     iiret
2459                                    ;
2460                                    ; 22/08/2014
2461                                    ;mov   al, 20h ; END OF INTERRUPT COMMAND TO 8259
2462                                    ;out   20h, al ; 8259 PORT
2463                                    ;
2464                                    ;pop   es
2465                                    ;pop   ds
2466                                    ;pop   edi
2467                                    ;pop   esi
2468                                    ;pop   ebx
2469                                    ;pop   eax
2470                                    ;iret
2471
2472                                    ; 02/04/2015
2473                                    ; 25/08/2014
2474                                exc0:
2475 00000C1E 6A00                          push       dword 0
2476 00000C20 E990000000                    jmp     cpu_except
2477                                exc1:
2478 00000C25 6A01                          push       dword 1
2479 00000C27 E989000000                    jmp     cpu_except
2480                                exc2:
2481 00000C2C 6A02                          push       dword 2
2482 00000C2E E982000000                    jmp     cpu_except
2483                                exc3:
2484 00000C33 6A03                          push       dword 3
2485 00000C35 EB7E                          jmp     cpu_except
2486                                exc4:
2487 00000C37 6A04                          push       dword 4
2488 00000C39 EB7A                          jmp     cpu_except
2489                                exc5:
2490 00000C3B 6A05                          push       dword 5
2491 00000C3D EB76                          jmp     cpu_except
2492                                exc6:
2493 00000C3F 6A06                          push       dword 6
2494 00000C41 EB72                          jmp     cpu_except
2495                                exc7:
2496 00000C43 6A07                          push       dword 7
2497 00000C45 EB6E                          jmp     cpu_except
2498                                exc8:
```

```
                                        ; [esp] = Error code
00000C47 6A08                             push      dword 8
00000C49 EB5C                             jmp     cpu_except_en
                                    exc9:
00000C4B 6A09                             push      dword 9
00000C4D EB66                             jmp     cpu_except
                                    exc10:
                                        ; [esp] = Error code
00000C4F 6A0A                             push      dword 10
00000C51 EB54                             jmp     cpu_except_en
                                    exc11:
                                        ; [esp] = Error code
00000C53 6A0B                             push      dword 11
00000C55 EB50                             jmp     cpu_except_en
                                    exc12:
                                        ; [esp] = Error code
00000C57 6A0C                             push      dword 12
00000C59 EB4C                             jmp     cpu_except_en
                                    exc13:
                                        ; [esp] = Error code
00000C5B 6A0D                             push      dword 13
00000C5D EB48                             jmp     cpu_except_en
                                    exc14:
                                        ; [esp] = Error code
00000C5F 6A0E                             push      dword 14
00000C61 EB44                         jmp     short cpu_except_en
                                    exc15:
00000C63 6A0F                             push      dword 15
00000C65 EB4E                             jmp     cpu_except
                                    exc16:
00000C67 6A10                             push      dword 16
00000C69 EB4A                             jmp     cpu_except
                                    exc17:
                                        ; [esp] = Error code
00000C6B 6A11                             push      dword 17
00000C6D EB38                         jmp     short cpu_except_en
                                    exc18:
00000C6F 6A12                             push      dword 18
00000C71 EB42                         jmp     short cpu_except
                                    exc19:
00000C73 6A13                             push      dword 19
00000C75 EB3E                         jmp     short cpu_except
                                    exc20:
00000C77 6A14                             push      dword 20
00000C79 EB3A                         jmp     short cpu_except
                                    exc21:
00000C7B 6A15                             push      dword 21
00000C7D EB36                         jmp     short cpu_except
                                    exc22:
00000C7F 6A16                             push      dword 22
00000C81 EB32                         jmp     short cpu_except
                                    exc23:
00000C83 6A17                             push      dword 23
00000C85 EB2E                         jmp     short cpu_except
                                    exc24:
00000C87 6A18                             push      dword 24
00000C89 EB2A                         jmp     short cpu_except
                                    exc25:
00000C8B 6A19                             push      dword 25
00000C8D EB26                         jmp     short cpu_except
                                    exc26:
00000C8F 6A1A                             push      dword 26
00000C91 EB22                         jmp     short cpu_except
                                    exc27:
00000C93 6A1B                             push      dword 27
00000C95 EB1E                         jmp     short cpu_except
                                    exc28:
00000C97 6A1C                             push      dword 28
00000C99 EB1A                         jmp     short cpu_except
                                    exc29:
00000C9B 6A1D                             push      dword 29
00000C9D EB16                         jmp     short cpu_except
                                    exc30:
00000C9F 6A1E                             push      dword 30
00000CA1 EB04                         jmp     short cpu_except_en
                                    exc31:
00000CA3 6A1F                             push      dword 31
00000CA5 EB0E                             jmp     short cpu_except

                                        ; 19/10/2015
                                        ; 19/09/2015
                                        ; 01/09/2015
                                        ; 28/08/2015
                                        ; 28/08/2014
                                    cpu_except_en:
00000CA7 87442404                       xchg    eax, [esp+4] ; Error code
00000CAB 36A3[48900100]                 mov     [ss:error_code], eax
00000CB1 58                             pop     eax  ; Exception number
00000CB2 870424                         xchg    eax, [esp]
                                                ; eax = eax before exception
                                                ; [esp] -> exception number
                                                ; [esp+4] -> EIP to return
                                        ; 22/01/2017
                                        ; 19/10/2015
                                        ; 19/09/2015
                                        ; 01/09/2015
                                        ; 28/08/2015
                                        ; 29/08/2014
                                        ; 28/08/2014
                                        ; 25/08/2014
                                        ; 21/08/2014
                                    cpu_except: ; CPU Exceptions
00000CB5 FC                             cld
00000CB6 870424                         xchg    eax, [esp]
                                                ; eax = Exception number
                                                ; [esp] = eax (before exception)
00000CB9 53                             push    ebx
00000CBA 56                             push    esi
00000CBB 57                             push    edi
00000CBC 1E                             push    ds
00000CBD 06                             push    es
                                        ; 28/08/2015
00000CBE 66BB1000                       mov     bx, KDATA
00000CC2 8EDB                           mov     ds, bx
00000CC4 8EC3                           mov     es, bx
00000CC6 0F20DB                         mov     ebx, cr3
00000CC9 53                             push    ebx ; (*) page directory
                                        ; 19/10/2015
00000CCA FC                             cld
                                        ; 25/03/2015
00000CCB 8B1D[80760100]                 mov     ebx, [k_page_dir]
00000CD1 0F22DB                         mov     cr3, ebx
                                        ; 28/08/2015
00000CD4 83F80E                         cmp     eax, 0Eh ; 14, PAGE FAULT
```

```
2623 00000CD7 7510                      jne     short cpu_except_nfp
2624 00000CD9 E8EF4D0000                call    page_fault_handler
2625 00000CDE 21C0                      and     eax, eax
2626                                    ;jz     iiretp ; 01/09/2015
2627                                    ; 24/07/2022
2628 00000CE0 7505                      jnz     short cpu_except_pf
2629 00000CE2 E9FD000000                jmp     iiretp
2630                                cpu_except_pf:     ; 24/07/2022
2631 00000CE7 B00E                      mov     al, 0Eh ; 14
2632                                cpu_except_nfp:
2633                                    ; 23/08/2016
2634 00000CE9 803D[1E680000]03          cmp     byte [CRT_MODE], 3
2635 00000CF0 7409                      je      short cpu_except_mode_3
2636 00000CF2 50                        push    eax
2637 00000CF3 B003                      mov     al, 3
2638 00000CF5 E88F0E0000                call    _set_mode
2639 00000CFA 58                        pop     eax
2640                                cpu_except_mode_3:
2641                                    ; 02/04/2015
2642 00000CFB BB[FD080000]              mov     ebx, hang
2643 00000D00 875C241C                  xchg    ebx, [esp+28]
2644                                            ; EIP (points to instruction which faults)
2645                                            ; New EIP (hang)
2646 00000D04 891D[4C900100]            mov     [FaultOffset], ebx
2647 00000D0A C744242008000000          mov     dword [esp+32], KCODE ; kernel's code segment
2648 00000D12 814C242400020000          or      dword [esp+36], 200h ; enable interrupts (set IF)
2649                                    ;
2650 00000D1A 88C4                      mov     ah, al
2651 00000D1C 240F                      and     al, 0Fh
2652 00000D1E 3C09                      cmp     al, 9
2653 00000D20 7602                      jna     short h1ok
2654 00000D22 0407                      add     al, 'A'-':'
2655                                h1ok:
2656 00000D24 C0EC04                    shr     ah, 4
2657 00000D27 80FC09                    cmp     ah, 9
2658 00000D2A 7603                      jna     short h2ok
2659 00000D2C 80C407                    add     ah, 'A'-':'
2660                                h2ok:
2661 00000D2F 86C4                      xchg    ah, al
2662 00000D31 66053030                  add     ax, '00'
2663 00000D35 66A3[18380100]            mov     [excnstr], ax
2664                                    ;
2665                                    ; 29/08/2014
2666 00000D3B A1[4C900100]              mov     eax, [FaultOffset]
2667 00000D40 51                        push    ecx
2668 00000D41 52                        push    edx
2669 00000D42 89E3                      mov     ebx, esp
2670                                    ; 28/08/2015
2671 00000D44 B910000000                mov     ecx, 16  ; divisor value to convert binary number
2672                                                     ; to hexadecimal string
2673                                    ;mov    ecx, 10    ; divisor to convert
2674                                                       ; binary number to decimal string
2675                                b2d1:
2676 00000D49 31D2                      xor     edx, edx
2677 00000D4B F7F1                      div     ecx
2678                                    ;push   dx
2679                                    ; 18/04/2021
2680 00000D4D 52                        push    edx
2681 00000D4E 39C8                      cmp     eax, ecx
2682 00000D50 73F7                      jnb     short b2d1
2683 00000D52 BF[23380100]              mov     edi, EIPstr ; EIP value
2684                                                        ; points to instruction which faults
2685                                    ; 28/08/2015
2686 00000D57 89C2                      mov     edx, eax
2687                                b2d2:
2688                                    ;add    al, '0'
2689 00000D59 8A82[09420000]            mov     al, [edx+hexchrs]
2690 00000D5F AA                        stosb               ; write hexadecimal digit to its place
2691 00000D60 39E3                      cmp     ebx, esp
2692 00000D62 7605                      jna     short b2d3
2693                                    ;pop    ax
2694                                    ; 18/04/2021
2695 00000D64 58                        pop     eax
2696 00000D65 88C2                      mov     dl, al
2697 00000D67 EBF0                      jmp     short b2d2
2698                                b2d3:
2699 00000D69 B068                      mov     al, 'h' ; 28/08/2015
2700 00000D6B AA                        stosb
2701 00000D6C B020                      mov     al, 20h     ; space
2702 00000D6E AA                        stosb
2703 00000D6F 30C0                      xor     al, al      ; to do it an ASCIIZ string
2704 00000D71 AA                        stosb
2705                                    ;
2706 00000D72 5A                        pop     edx
2707 00000D73 59                        pop     ecx
2708                                    ;
2709 00000D74 B44F                      mov     ah, 4Fh ; red (4) background,
2710                                                    ; white (F) forecolor
2711 00000D76 BE[08380100]              mov     esi, exc_msg ; message offset
2712                                    ;
2713                                    ; 20/01/2017 (!cpu exception!)
2714                                    ;
2715 00000D7B 8105[C0350100]A000-           add dword [scr_row], 0A0h
2715 00000D83 0000
2716 00000D85 8B3D[C0350100]                mov edi, [scr_row]
2717                                    ;
2718 00000D8B C605[288E0100]00          mov     byte [sysflg], 0  ; system mode
2719 00000D92 FB                            sti
2720                                    ;
2721 00000D93 E89FFBFFFF                call    printk
2722                                    ;
2723 00000D98 B410                      mov     ah, 10h
2724 00000D9A E87D010000                call    int16h ; getc
2725                                    ;
2726 00000D9F B003                      mov     al, 3
2727 00000DA1 E8E30D0000                call    _set_mode
2728                                    ;
2729                                    ;mov    eax, 1
2730                                    ; 30/11/2023
2731 00000DA6 31C0                      xor     eax, eax
2732 00000DA8 40                        inc     eax
2733                                    ; eax = 1
2734 00000DA9 E90EBE0000                jmp     sysexit ; terminate process !!!
2735
2736                                    ; 22/01/2017
2737                                    ; 18/04/2016
2738                                    ; 28/08/2015
2739                                    ; 23/02/2015
2740                                    ; 20/08/2014
2741                                ignore_int:
2742 00000DAE 50                        push    eax
2743 00000DAF 53                        push    ebx ; 23/02/2015
2744 00000DB0 56                        push    esi
2745 00000DB1 57                        push    edi
```

```
2746 00000DB2 1E                        push    ds
2747 00000DB3 06                        push    es
2748                                    ; 18/04/2016
2749 00000DB4 66B81000                  mov     ax, KDATA
2750 00000DB8 8ED8                      mov     ds, ax
2751 00000DBA 8EC0                      mov     es, ax
2752                                    ; 28/08/2015
2753 00000DBC 0F20D8                    mov     eax, cr3
2754 00000DBF 50                        push    eax ; (*) page directory
2755                                    ;
2756 00000DC0 B467                      mov     ah, 67h ; brown (6) background,
2757                                                    ; light gray (7) forecolor
2758 00000DC2 BE[D0360100]              mov     esi, int_msg ; message offset
2759                                piemsg:
2760                                        ; 27/08/2014
2761 00000DC7 8105[C0350100]A000-           add     dword [scr_row], 0A0h
2761 00000DCF 0000
2762 00000DD1 8B3D[C0350100]                mov     edi, [scr_row]
2763                                        ;
2764 00000DD7 E85BFBFFFF                call    printk
2765                                    ;
2766                                    ; 23/02/2015
2767 00000DDC B020                      mov     al, 20h  ; END OF INTERRUPT COMMAND TO
2768 00000DDE E6A0                      out     0A0h, al ; the 2nd 8259
2769                                    ; 22/08/2014
2770 00000DE0 B020                      mov     al, 20h ; END OF INTERRUPT COMMAND TO 8259
2771 00000DE2 E620                      out     20h, al ; 8259 PORT
2772                                iiretp:
2773                                    ; 22/01/2017
2774                                    ; 01/09/2015
2775                                    ; 28/08/2015
2776 00000DE4 58                        pop     eax ; (*) page directory
2777 00000DE5 0F22D8                    mov     cr3, eax
2778                                iiret:
2779 00000DE8 07                        pop     es
2780 00000DE9 1F                        pop     ds
2781 00000DEA 5F                        pop     edi
2782 00000DEB 5E                        pop     esi
2783 00000DEC 5B                        pop     ebx ; 29/08/2014
2784 00000DED 58                        pop     eax
2785 00000DEE CF                        iretd
2786
2787                                    ; 23/05/2016
2788                                    ; 22/08/2014
2789                                    ; IBM PC/AT BIOS source code ----- 10/06/85 (bios.asm)
2790                                    ; (INT 1Ah)
2791                                    ;; Linux (v0.12) source code (main.c) by Linus Torvalds (1991)
2792                                time_of_day:
2793 00000DEF E829570000                call    UPD_IPR                 ; WAIT TILL UPDATE NOT IN PROGRESS
2794 00000DF4 726F                          jc      short time_of_day_retn ; 23/05/2016
2795 00000DF6 B000                      mov     al, CMOS_SECONDS
2796 00000DF8 E856570000                call    CMOS_READ
2797 00000DFD A2[F0760100]              mov     [time_seconds], al
2798 00000E02 B002                      mov     al, CMOS_MINUTES
2799 00000E04 E84A570000                call    CMOS_READ
2800 00000E09 A2[F1760100]              mov     [time_minutes], al
2801 00000E0E B004                      mov     al, CMOS_HOURS
2802 00000E10 E83E570000                call    CMOS_READ
2803 00000E15 A2[F2760100]                  mov     [time_hours], al
2804 00000E1A B006                      mov     al, CMOS_DAY_WEEK
2805 00000E1C E832570000                call    CMOS_READ
2806 00000E21 A2[F3760100]              mov     [date_wday], al
2807 00000E26 B007                      mov     al, CMOS_DAY_MONTH
2808 00000E28 E826570000                call    CMOS_READ
2809 00000E2D A2[F4760100]              mov     [date_day], al
2810 00000E32 B008                      mov     al, CMOS_MONTH
2811 00000E34 E81A570000                call    CMOS_READ
2812 00000E39 A2[F5760100]              mov     [date_month], al
2813 00000E3E B009                      mov     al, CMOS_YEAR
2814 00000E40 E80E570000                call    CMOS_READ
2815 00000E45 A2[F6760100]              mov     [date_year], al
2816 00000E4A B032                      mov     al, CMOS_CENTURY
2817 00000E4C E802570000                call    CMOS_READ
2818 00000E51 A2[F7760100]              mov     [date_century], al
2819                                    ;
2820 00000E56 B000                      mov     al, CMOS_SECONDS
2821 00000E58 E8F6560000                call    CMOS_READ
2822 00000E5D 3A05[F0760100]            cmp     al, [time_seconds]
2823 00000E63 758A                      jne     short time_of_day
2824
2825                                time_of_day_retn:
2826 00000E65 C3                        retn
2827
2828                                    ; 15/01/2017
2829                                    ; 10/06/2016
2830                                    ; 07/06/2016
2831                                    ; 06/06/2016
2832                                    ; 23/05/2016
2833                                rtc_p:
2834 00000E66 B101                      mov     cl, 1 ; 15/01/2017
2835 00000E68 EB02                      jmp     short rtc_p0
2836                                u_timer:
2837                                    ; Timer Events with 18.2 Hz Timer Ticks
2838                                    ; (and also timer events with RTC seconds)
2839 00000E6A 28C9                      sub     cl, cl ; mov cl, 0 ; 15/01/2017
2840                                rtc_p0:
2841                                    ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
2842                                    ; Major Modification:
2843                                    ; Check and Perform Timer Events (for RTC)
2844                                    ; 25/08/2014 - 07/09/2014
2845                                    ; Retro UNIX 386 v1:
2846                                    ; Print Real Time Clock content
2847
2848                                    ; 15/01/2017
2849 00000E6C 880D[10830100]            mov     byte [priority], cl ; 0 or 1 (not 2)
2850 00000E72 8A2D[13830100]            mov     ch, [timer_events]
2851 00000E78 20ED                      and     ch, ch
2852 00000E7A 7420                      jz      short rtc_p3
2853
2854 00000E7C BE[488F0100]              mov     esi, timer_set  ; beginning address of
2855                                                             ; timer events space
2856                                rtc_p1:
2857 00000E81 8B06                      mov     eax, [esi]
2858 00000E83 20C0                      and     al, al ; 0 = free, >0 = process no.
2859 00000E85 7416                      jz      short rtc_p4
2860                                    ;
2861 00000E87 C1C810                    ror     eax, 16
2862                                    ; ah = response value, al = interrupt type
2863                                    ; 15/01/2017
2864                                    ; cl = interrupt source
2865                                    ;       1 = RTC, 0 = PIT
2866 00000E8A 38C8                      cmp     al, cl
2867 00000E8C 750A                      jne     short rtc_p2 ; not as requested or undefined !
2868 00000E8E 3C01                      cmp     al, 1 ; 1 ; RTC interrupt ?
```

```
00000E90 7410                       je      short rtc_p5 ; yes, check for response
                                    ; 06/06/2016 - 18.2 Hz Timer Ticks
00000E92 836E080A                   sub     dword [esi+8], 10 ; 1 tick = 10
00000E96 7613                       jna     short rtc_p6  ; continue for responding
                                rtc_p2:
                                    ; 15/01/2017 (cl -> ch)
                                    ; 07/06/2016
00000E98 FECD                       dec     ch    ; remain count of timer events
00000E9A 7501                       jnz     short rtc_p4
                                rtc_p3:
00000E9C C3                         retn
                                rtc_p4:
                                    ;cmp    esi, timer_set + 240 ; 15*16 (last event)
                                    ;jnb    short rtc_p3 ; end of timer event space
00000E9D 83C610                     add     esi, 16 ; next timer event
00000EA0 EBDF                       jmp     short rtc_p1
                                rtc_p5:
                                    ; current timer count ; 06/06/2016 (182)
00000EA2 816E08B6000000             sub     dword [esi+8], 182 ; 1 second (10*18.2)
00000EA9 77ED                       ja      short rtc_p2  ; check for the next
                                rtc_p6:
                                    ; it is the time of response!
00000EAB 8B5E04                     mov     ebx, [esi+4] ; set (count limit) value
00000EAE 895E08                     mov     [esi+8], ebx ; reset count down value
                                                         ; to count limit
                                    ; 19/12/2016
                                    ; 10/12/2016 - timer callback modification
00000EB1 8B7E0C                     mov     edi, [esi+12] ; response (or callback) address
00000EB4 807E0100                   cmp     byte [esi+1], 0 ; >0 = callback
00000EB8 762A                       jna     short rtc_p8

                                    ; timer callback !
00000EBA 0FB61E                     movzx   ebx, byte [esi] ; process number (>0)
00000EBD 89D8                       mov     eax, ebx
00000EBF C0E302                     shl     bl, 2 ; *4
00000EC2 899B[C48D0100]             mov     [ebx+p.tcb-4], edi ; user's callback service addr
00000EC8 3A05[858E0100]             cmp     al, [u.uno]
00000ECE 7521                       jne     short rtc_p9
00000ED0 893D[A88E0100]             mov     [u.tcb], edi
                                rtc_p7:
                                    ; 15/01/2017
00000ED6 B002                       mov     al, 2
00000ED8 A2[10830100]               mov     [priority], al ; 2
                                    ; 10/01/2017
                                    ;mov    byte [u.pri], 2
00000EDD A2[7E8E0100]               mov     [u.pri], al ; 2
00000EE2 EBB4                       jmp     short rtc_p2
                                rtc_p8:
                                    ; response address is physical address of
                                    ; the program's response (signal return) byte
                                    ; 06/06/2016
                                    ;mov    edi, [esi+12] ; response address
00000EE4 8827                       mov     [edi], ah     ; response value
                                    ;
00000EE6 C1C010                     rol     eax, 16
                                    ; 15/01/2017
00000EE9 3A05[858E0100]             cmp     al, [u.uno] ; running process ?
00000EEF 74E5                       je      short rtc_p7
                                rtc_p9:
                                    ; al = process number  ; 10/06/2016
00000EF1 B202                       mov     dl, 2 ; priority, 2 = event (high)
00000EF3 E8D20B0100                 call    set_run_sequence ; 19/05/2016
00000EF8 EB9E                       jmp     short rtc_p2 ; 10/06/2016

                                ; Default IRQ 7 handler against spurious IRQs (from master PIC)
                                ; 25/02/2015 (source: http://wiki.osdev.org/8259_PIC)
                                default_irq7:
                                    ;push   ax
                                    ; 18/04/2021
00000EFA 50                         push    eax
00000EFB B00B                       mov     al, 0Bh  ; In-Service register
00000EFD E620                       out     20h, al
                                        jmp short $+2
00000EFF EB00                       jmp short $+2
00000F01 EB00                       in      al, 20h
00000F03 E420                       and     al, 80h ; bit 7 (is it real IRQ 7 or fake?)
00000F05 2480                           jz      short irq7_iret ; Fake (spurious) IRQ, do not send EOI
00000F07 7404                           mov     al, 20h ; EOI
00000F09 B020                       out     20h, al
00000F0B E620                   irq7_iret:
                                    ;pop    ax
                                    ; 18/04/2021
00000F0D 58                         pop     eax
00000F0E 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)
                                    ;
00000F0F D410                       db      0D4h, 10h       ; Undocumented inst. AAM
                                                            ; AH = AL / 10h
                                                            ; AL = AL MOD 10h
00000F11 660D3030                   or      ax, '00'        ; Make it ASCII based

00000F15 86C4                           xchg        ah, al

00000F17 C3                         retn

                                ; 15/12/2020
                                real_mem_16m_64k:
00000F18 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:
00000F1A 0000                       dw      0       ; HW of default LFB addr (for mode 118h)


                                %include 'keyboard.s' ; 07/03/2015
                            <1> ; ****************************************************************************
                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.5 - keyboard.s
                            <1> ; ----------------------------------------------------------------------------
                            <1> ; Last Update: 07/08/2022 (Previous: 12/04/2021)
```

```
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Beginning: 17/01/2016
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Assembler: NASM version 2.15 (trdos386.s)
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Turkish Rational DOS
                        <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
                        <1> ;
                        <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
                        <1> ; keyboard.inc (17/10/2015)
                        <1> ;
                        <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
                        <1> ; ****************************************************************************
                        <1>
                        <1> ; Ref: Retro UNIX 386 v1.2 - keyboard.s - 11/06/2022
                        <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:
00000F1C 9C             <1>     pushfd  ; 28/08/2014
00000F1D 0E             <1>     push    cs
00000F1E E826000000     <1>     call    KEYBOARD_IO_1 ; getc_int
00000F23 C3             <1>     retn
                        <1>
                        <1> ; 24/07/2022 - TRDOS 386 v2.0.5
                        <1>
                        <1>     ;----- SHIFT STATUS
                        <1> _K3E:                                       ; GET THE EXTENDED SHIFT STATUS FLAGS
00000F24 8A25[EA670000] <1>     mov     ah, [KB_FLAG_1]                  ; GET SYSTEM SHIFT KEY STATUS
00000F2A 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
00000F2D C0E405         <1>         shl ah, 5
00000F30 A0[EA670000]   <1>     mov     al, [KB_FLAG_1]                 ; GET SYSTEM SHIFT STATES BACK
00000F35 2473           <1>     and     al, 01110011b           ; ELIMINATE SYS SHIFT, HOLD_STATE AND INS_SHIFT
00000F37 08C4           <1>     or      ah, al                    ; MERGE REMAINING BITS INTO AH
00000F39 A0[EC670000]   <1>     mov     al, [KB_FLAG_3]                 ; GET RIGHT CTL AND ALT
00000F3E 240C           <1>     and     al, 00001100b           ; ELIMINATE LC_E0 AND LC_E1
00000F40 08C4           <1>     or      ah, al                  ; OR THE SHIFT FLAGS TOGETHER
                        <1> _K3:
00000F42 A0[E9670000]   <1>     mov     al, [KB_FLAG]           ; GET THE SHIFT STATUS FLAGS
                        <1>     ; 24/07/2022
00000F47 EB38           <1>     jmp     short _KIO_EXIT                 ; RETURN TO CALLER
                        <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> ; 07/08/2022
                        <1> ; 24/07/2022 - TRDOS 386 v2.0.5
                        <1> ; 12/04/2021 - TRDOS 386 v2.0.3 (32 bit push/pop)
                        <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
00000F49 80642408BE     <1>         and     byte [esp+8], 10111110b ; clear zero flag and cary flag
                        <1>     ;
00000F4E 1E             <1>     push    ds                      ; SAVE CURRENT DS
00000F4F 53             <1>     push    ebx                     ; SAVE BX TEMPORARILY
                        <1>     ;push   ecx                     ; SAVE CX TEMPORARILY
00000F50 66BB1000       <1>         mov     bx, KDATA
00000F54 8EDB           <1>     mov     ds, bx                  ; PUT SEGMENT VALUE OF DATA AREA INTO DS
                        <1>     ; 14/01/2017
00000F56 8B1C24         <1>     mov     ebx, [esp]
                        <1>     ;; 15/01/2017
                        <1>     ; 02/01/2017
                        <1>     ;;mov   byte [intflg], 32h      ; keyboard interrupt
00000F59 FB             <1>     sti
                        <1>     ;
00000F5A 08E4           <1>     or      ah, ah                  ; CHECK FOR (AH)= 00H
00000F5C 7433           <1>     jz      short _K1               ; ASCII_READ
00000F5E FECC           <1>     dec     ah                        ; CHECK FOR (AH)= 01H
00000F60 744C           <1>         jz      short _K2                   ; ASCII_STATUS
00000F62 FECC           <1>     dec     ah                      ; CHECK FOR (AH)= 02H
00000F64 74DC           <1>         jz  short _K3               ; SHIFT STATUS
00000F66 FECC           <1>     dec     ah                      ; CHECK FOR (AH)= 03H
00000F68 746F           <1>         jz      short _K300             ; SET TYPAMATIC RATE/DELAY
00000F6A 80EC02         <1>     sub     ah, 2                   ; CHECK FOR (AH)= 05H
                        <1>         ;jz short _K500             ; KEYBOARD WRITE
                        <1>     ; 07/08/2022
00000F6D 7505           <1>     jnz     short _KIO1
00000F6F E988000000     <1>     jmp     _K500
                        <1> _KIO1:
00000F74 80EC0B         <1>     sub     ah, 11                  ; AH =  10H
00000F77 740C           <1>     jz      short _K1E              ; EXTENDED ASCII READ
00000F79 FECC           <1>     dec     ah                      ; CHECK FOR (AH)= 11H
00000F7B 7422           <1>     jz      short _K2E              ; EXTENDED_ASCII_STATUS
00000F7D FECC           <1>     dec     ah                      ; CHECK FOR (AH)= 12H
00000F7F 74A3           <1>     jz      short _K3E              ; EXTENDED_SHIFT_STATUS
                        <1> _KIO_EXIT:
                        <1>     ; 02/01/2017
00000F81 FA             <1>     cli
                        <1>     ;;mov   byte [intflg], 0 ;; 15/01/2017
                        <1>     ;
                        <1>     ;pop    ecx                     ; RECOVER REGISTER
00000F82 5B             <1>     pop     ebx                     ; RECOVER REGISTER
00000F83 1F             <1>     pop     ds                      ; RECOVER SEGMENT
00000F84 CF             <1>     iretd                           ; INVALID COMMAND, EXIT
                        <1>
                        <1> ; 24/07/2022
                        <1> ;
                        <1> ;   ;----- SHIFT STATUS
                        <1> ;_K3E:                                      ; GET THE EXTENDED SHIFT STATUS FLAGS
                        <1> ;   mov     ah, [KB_FLAG_1]                 ; GET SYSTEM SHIFT KEY STATUS
                        <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
                        <1> ;       shl ah, 5
                        <1> ;   mov     al, [KB_FLAG_1]                 ; GET SYSTEM SHIFT STATES BACK
                        <1> ;   and     al, 01110011b           ; ELIMINATE SYS SHIFT, HOLD_STATE AND INS_SHIFT
                        <1> ;   or      ah, al                    ; MERGE REMAINING BITS INTO AH
                        <1> ;   mov     al, [KB_FLAG_3]                 ; GET RIGHT CTL AND ALT
                        <1> ;   and     al, 00001100b           ; ELIMINATE LC_E0 AND LC_E1
                        <1> ;   or      ah, al                  ; OR THE SHIFT FLAGS TOGETHER
                        <1> ;_K3:
                        <1> ;   mov     al, [KB_FLAG]           ; GET THE SHIFT STATUS FLAGS
                        <1> ;   ; 24/07/2022
                        <1> ;   jmp     short _KIO_EXIT                 ; RETURN TO CALLER
                        <1>
                        <1>     ;----- ASCII CHARACTER
                        <1> _K1E:
00000F85 E89F000000     <1>     call    _K1S                    ; GET A CHARACTER FROM THE BUFFER (EXTENDED)
00000F8A E812010000     <1>     call    _KIO_E_XLAT             ; ROUTINE TO XLATE FOR EXTENDED CALLS
00000F8F EBF0           <1>     jmp     short _KIO_EXIT           ; GIVE IT TO THE CALLER
                        <1> _K1:
00000F91 E893000000     <1>     call    _K1S                    ; GET A CHARACTER FROM THE BUFFER
00000F96 E811010000     <1>     call    _KIO_S_XLAT             ; ROUTINE TO XLATE FOR STANDARD CALLS
00000F9B 72F4           <1>     jc      short _K1               ; CARRY SET MEANS TROW CODE AWAY
                        <1> _K1A:
00000F9D EBE2           <1>     jmp     short _KIO_EXIT           ; RETURN TO CALLER
                        <1>
                        <1>     ;----- ASCII STATUS
                        <1> _K2E:
00000F9F E8D0000000     <1>     call    _K2S                    ; TEST FOR CHARACTER IN BUFFER (EXTENDED)
00000FA4 7420           <1>     jz      short _K2B              ; RETURN IF BUFFER EMPTY
00000FA6 9C             <1>     pushf                           ; SAVE ZF FROM TEST
00000FA7 E8F5000000     <1>     call    _KIO_E_XLAT             ; ROUTINE TO XLATE FOR EXTENDED CALLS
00000FAC EB17           <1>     jmp     short _K2A               ; GIVE IT TO THE CALLER
```

```
                        <1> _K2:
00000FAE E8C1000000      <1>     call    _K2S                    ; TEST FOR CHARACTER IN BUFFER
00000FB3 7411            <1>     jz      short _K2B              ; RETURN IF BUFFER EMPTY
00000FB5 9C              <1>     pushf                           ; SAVE ZF FROM TEST
00000FB6 E8F1000000      <1>     call    _KIO_S_XLAT             ; ROUTINE TO XLATE FOR STANDARD CALLS
00000FBB 7308            <1>     jnc     short _K2A               ; CARRY CLEAR MEANS PASS VALID CODE
00000FBD 9D              <1>     popf                            ; INVALID CODE FOR THIS TYPE OF CALL
00000FBE E866000000      <1>     call    _K1S                    ; THROW THE CHARACTER AWAY
00000FC3 EBE9            <1>     jmp     short _K2               ; GO LOOK FOR NEXT CHAR, IF ANY
                        <1> _K2A:
00000FC5 9D              <1>     popf                            ; RESTORE ZF FROM TEST
                        <1> _K2B:
                        <1>     ; 02/01/2017
00000FC6 FA              <1>     cli
                        <1>     ;; mov  byte [intflg], 0 ;; 15/01/2017
                        <1>     ;
                        <1>     ;pop    ecx                     ; RECOVER REGISTER
00000FC7 5B              <1>     pop     ebx                     ; RECOVER REGISTER
00000FC8 1F              <1>     pop     ds                      ; RECOVER SEGMENT
                        <1>     ; (*) 29/05/2016
                        <1>     ; (*) retf 4                    ; THROW AWAY (e)FLAGS
00000FC9 7208            <1>     jc      short _k2d
00000FCB 7505            <1>     jnz     short _k2c
00000FCD 804C240840      <1>     or      byte [esp+8], 01000000b       ; set zero flag bit of eflags register
                        <1> _k2c:
00000FD2 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
00000FD3 804C240801      <1>     or      byte [esp+8], 1  ; set carry bit of eflags register
                        <1>     ; [esp+12] = ESP (user)
                        <1>     ; [esp+16] = SS (User)
00000FD8 CF              <1>     iretd
                        <1>
                        <1>     ; (*) 29/05/2016 - 'retf 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 SP by 4 plus the immediate offset if it exists;
                        <1>     ;    Load SS:SP from stack;
                        <1>     ; FI;
                        <1>     ;
                        <1>     ; //
                        <1>
                        <1>     ; 24/07/2022
                        <1>     ;----- SET TYPAMATIC RATE AND DELAY
                        <1> _K300:
00000FD9 3C05            <1>     cmp     al, 5                   ; CORRECT FUNCTION CALL?
00000FDB 75A4            <1>     jne     short _KIO_EXIT                 ; NO, RETURN
00000FDD F6C3E0          <1>     test    bl, 0E0h                ; TEST FOR OUT-OF-RANGE RATE
00000FE0 759F            <1>     jnz     short _KIO_EXIT                 ; RETURN IF SO
00000FE2 F6C7FC          <1>     test    bh, 0FCh                ; TEST FOR OUT-OF-RANGE DELAY
00000FE5 759A            <1>     jnz     short _KIO_EXIT                 ; RETURN IF SO
00000FE7 B0F3            <1>     mov     al, KB_TYPA_RD          ; COMMAND FOR TYPAMATIC RATE/DELAY
00000FE9 E875060000      <1>     call    SND_DATA                ; SEND TO KEYBOARD
                        <1>     ;mov    cx, 5                   ; SHIFT COUNT
                        <1>     ;shl    bh, cl                  ; SHIFT DELAY OVER
00000FEE C0E705          <1>     shl     bh, 5
00000FF1 88D8            <1>     mov     al, bl                  ; PUT IN RATE
00000FF3 08F8            <1>     or      al, bh                  ; AND DELAY
00000FF5 E869060000      <1>     call    SND_DATA                ; SEND TO KEYBOARD
00000FFA EB85            <1>         jmp     _KIO_EXIT                ; RETURN TO CALLER
                        <1>
                        <1>     ;----- WRITE TO KEYBOARD BUFFER
                        <1> _K500:
00000FFC 56              <1>     push    esi                     ; SAVE SI (esi)
00000FFD FA              <1>     cli                             ;
00000FFE 8B1D[FA670000]  <1>             mov     ebx, [BUFFER_TAIL]     ; GET THE 'IN TO' POINTER TO THE BUFFER
00001004 89DE            <1>     mov     esi, ebx                ; SAVE A COPY IN CASE BUFFER NOT FULL
00001006 E8D1000000      <1>     call    _K4                     ; BUMP THE POINTER TO SEE IF BUFFER IS FULL
0000100B 3B1D[F6670000]  <1>     cmp     ebx, [BUFFER_HEAD]      ; WILL THE BUFFER OVERRUN IF WE STORE THIS?
00001011 740D            <1>     je      short _K502             ; YES - INFORM CALLER OF ERROR
00001013 66890E          <1>     mov     [esi], cx               ; NO - PUT ASCII/SCAN CODE INTO BUFFER
00001016 891D[FA670000]  <1>     mov     [BUFFER_TAIL], ebx      ; ADJUST 'IN TO' POINTER TO REFLECT CHANGE
0000101C 28C0            <1>     sub     al, al                  ; TELL CALLER THAT OPERATION WAS SUCCESSFUL
0000101E EB02            <1>     jmp     short _K504             ; SUB INSTRUCTION ALSO RESETS CARRY FLAG
                        <1> _K502:
00001020 B001            <1>     mov     al, 01h                 ; BUFFER FULL INDICATION
                        <1> _K504:
00001022 FB              <1>     sti
00001023 5E              <1>     pop     esi                     ; RECOVER SI (esi)
00001024 E958FFFFFF      <1>         jmp     _KIO_EXIT                ; RETURN TO CALLER WITH STATUS IN AL
                        <1>
                        <1>     ;----- READ THE KEY TO FIGURE OUT WHAT TO DO -----
                        <1> _K1S:
00001029 FA              <1>     cli     ; 03/12/2014
0000102A 8B1D[F6670000]  <1>         mov     ebx, [BUFFER_HEAD]       ; GET POINTER TO HEAD OF BUFFER
00001030 3B1D[FA670000]  <1>         cmp     ebx, [BUFFER_TAIL]       ; TEST END OF BUFFER
                        <1>     ;jne    short _K1U              ; IF ANYTHING IN BUFFER SKIP INTERRUPT
00001036 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
00001038 FB              <1>     sti                             ; INTERRUPTS BACK ON DURING LOOP
00001039 90              <1>     nop                             ; ALLOW AN INTERRUPT TO OCCUR
                        <1> _K1U:
0000103A FA              <1>     cli                             ; INTERRUPTS BACK OFF
0000103B 8B1D[F6670000]  <1>         mov        ebx, [BUFFER_HEAD]     ; GET POINTER TO HEAD OF BUFFER
00001041 3B1D[FA670000]  <1>         cmp     ebx, [BUFFER_TAIL]       ; TEST END OF BUFFER
                        <1> _k1x:
00001047 53              <1>     push    ebx                     ; SAVE ADDRESS
00001048 9C              <1>     pushf                           ; SAVE FLAGS
00001049 E8C9060000      <1>     call    MAKE_LED                ; GO GET MODE INDICATOR DATA BYTE
0000104E 8A1D[EB670000]  <1>     mov     bl, [KB_FLAG_2]         ; GET PREVIOUS BITS
00001054 30C3            <1>     xor     bl, al                  ; SEE IF ANY DIFFERENT
00001056 80E307          <1>     and     bl, 07h ; KB_LEDS       ; ISOLATE INDICATOR BITS
00001059 7406            <1>     jz      short _K1V              ; IF NO CHANGE BYPASS UPDATE
0000105B E863060000      <1>     call    SND_LED1
00001060 FA              <1>     cli                             ; DISABLE INTERRUPTS
```

```
                        <1> _K1V:
00001061 9D             <1>     popf                            ; RESTORE FLAGS
00001062 5B             <1>     pop     ebx                     ; RESTORE ADDRESS
00001063 74D3           <1>         je      short _K1T              ; LOOP UNTIL SOMETHING IN BUFFER
                        <1>     ;
00001065 668B03         <1>     mov     ax, [ebx]               ; GET SCAN CODE AND ASCII CODE
00001068 E86F000000     <1>         call    _K4                     ; MOVE POINTER TO NEXT POSITION
0000106D 891D[F6670000] <1>         mov     [BUFFER_HEAD], ebx      ; STORE VALUE IN VARIABLE
00001073 C3             <1>     retn                            ; RETURN
                        <1>
                        <1>     ;----- READ THE KEY TO SEE IF ONE IS PRESENT -----
                        <1> _K2S:
00001074 FA             <1>     cli                             ; INTERRUPTS OFF
00001075 8B1D[F6670000] <1>         mov     ebx, [BUFFER_HEAD]      ; GET HEAD POINTER
0000107B 3B1D[FA670000] <1>         cmp     ebx, [BUFFER_TAIL]      ; IF EQUAL (Z=1) THEN NOTHING THERE
00001081 668B03         <1>     mov     ax, [ebx]
00001084 9C             <1>     pushf                           ; SAVE FLAGS
                        <1>     ;push   ax                      ; SAVE CODE
                        <1>     ; 12/04/2021
00001085 50             <1>     push    eax
00001086 E88C060000     <1>     call    MAKE_LED                ; GO GET MODE INDICATOR DATA BYTE
0000108B 8A1D[EB670000] <1>     mov     bl, [KB_FLAG_2]         ; GET PREVIOUS BITS
00001091 30C3           <1>     xor     bl, al                  ; SEE IF ANY DIFFERENT
00001093 80E307         <1>     and     bl, 07h ; KB_LEDS       ; ISOLATE INDICATOR BITS
00001096 7405           <1>     jz      short _K2T              ; IF NO CHANGE BYPASS UPDATE
00001098 E80F060000     <1>     call    SND_LED                 ; GO TURN ON MODE INDICATORS
                        <1> _K2T:
                        <1>     ;pop    ax                      ; RESTORE CODE
                        <1>     ; 12/04/2021
0000109D 58             <1>     pop     eax
0000109E 9D             <1>     popf                            ; RESTORE FLAGS
0000109F FB             <1>     sti                             ; INTERRUPTS BACK ON
000010A0 C3             <1>     retn                            ; RETURN
                        <1>
                        <1>     ;----- ROUTINE TO TRANSLATE SCAN CODE PAIRS FOR EXTENDED CALLS -----
                        <1> _KIO_E_XLAT:
000010A1 3CF0           <1>     cmp     al, 0F0h                ; IS IT ONE OF THE FILL-INs?
000010A3 7506           <1>     jne     short _KIO_E_RET        ; NO, PASS IT ON
000010A5 08E4           <1>         or  ah, ah                  ; AH = 0 IS SPECIAL CASE
000010A7 7402           <1>         jz  short _KIO_E_RET        ; PASS THIS ON UNCHANGED
000010A9 30C0           <1>     xor     al, al                  ; OTHERWISE SET AL = 0
                        <1> _KIO_E_RET:
000010AB C3             <1>     retn                            ; GO BACK
                        <1>
                        <1>     ;----- ROUTINE TO TRANSLATE SCAN CODE PAIRS FOR STANDARD CALLS -----
                        <1> _KIO_S_XLAT:
000010AC 80FCE0         <1>     cmp     ah, 0E0h                ; IS IT KEYPAD ENTER OR / ?
000010AF 750F           <1>     jne     short _KIO_S2           ; NO, CONTINUE
000010B1 3C0D           <1>     cmp     al, 0Dh                 ; KEYPAD ENTER CODE?
000010B3 7408           <1>         je  short _KIO_S1           ; YES, MASSAGE A BIT
000010B5 3C0A           <1>     cmp     al, 0Ah                 ; CTRL KEYPAD ENTER CODE?
000010B7 7404           <1>         je  short _KIO_S1           ; YES, MASSAGE THE SAME
000010B9 B435           <1>     mov     ah, 35h                 ; NO, MUST BE KEYPAD /
                        <1> _kio_ret: ; 03/12/2014
000010BB F8             <1>     clc
000010BC C3             <1>     retn
                        <1>     ;jmp    short _KIO_USE          ; GIVE TO CALLER
                        <1> _KIO_S1:
000010BD B41C           <1>     mov     ah, 1Ch                 ; CONVERT TO COMPATIBLE OUTPUT
                        <1>     ;jmp    short _KIO_USE          ; GIVE TO CALLER
000010BF C3             <1>     retn
                        <1> _KIO_S2:
000010C0 80FC84         <1>     cmp     ah, 84h                 ; IS IT ONE OF EXTENDED ONES?
000010C3 7715           <1>     ja      short _KIO_DIS          ; YES, THROW AWAY AND GET ANOTHER CHAR
000010C5 3CF0           <1>     cmp     al, 0F0h                ; IS IT ONE OF THE FILL-INs?
000010C7 7506           <1>         jne short _KIO_S3           ; NO, TRY LAST TEST
000010C9 08E4           <1>     or      ah, ah                  ; AH = 0 IS SPECIAL CASE
000010CB 740C           <1>         jz  short _KIO_USE          ; PASS THIS ON UNCHANGED
000010CD EB0B           <1>     jmp     short _KIO_DIS          ; THROW AWAY THE REST
                        <1> _KIO_S3:
000010CF 3CE0           <1>     cmp     al, 0E0h                ; IS IT AN EXTENSION OF A PREVIOUS ONE?
                        <1>     ;jne    short _KIO_USE          ; NO, MUST BE A STANDARD CODE
000010D1 75E8           <1>     jne     short _kio_ret
000010D3 08E4           <1>     or      ah, ah                  ; AH = 0 IS SPECIAL CASE
000010D5 7402           <1>         jz  short _KIO_USE          ; JUMP IF AH = 0
000010D7 30C0           <1>     xor     al, al                  ; CONVERT TO COMPATIBLE OUTPUT
                        <1>     ;jmp    short _KIO_USE          ; PASS IT ON TO CALLER
                        <1> _KIO_USE:
                        <1>     ;clc                            ; CLEAR CARRY TO INDICATE GOOD CODE
000010D9 C3             <1>     retn                            ; RETURN
                        <1> _KIO_DIS:
000010DA F9             <1>     stc                             ; SET CARRY TO INDICATE DISCARD CODE
000010DB C3             <1>     retn                            ; RETURN
                        <1>
                        <1>     ;----- INCREMENT BUFFER POINTER ROUTINE -----
                        <1> _K4:
000010DC 43             <1>     inc     ebx
000010DD 43             <1>     inc     ebx                     ; MOVE TO NEXT WORD IN LIST
000010DE 3B1D[F2670000] <1>         cmp     ebx, [BUFFER_END]       ; AT END OF BUFFER?
                        <1>         ;jne    short _K5               ; NO, CONTINUE
000010E4 7206           <1>     jb      short _K5
000010E6 8B1D[EE670000] <1>         mov     ebx, [BUFFER_START]     ; YES, RESET TO BUFFER BEGINNING
                        <1> _K5:
000010EC C3             <1>     retn
                        <1>
                        <1> ; 20/02/2015
                        <1> ; 05/12/2014
                        <1> ; 26/08/2014
                        <1> ; KEYBOARD (HARDWARE) INTERRUPT -  IRQ LEVEL 1
                        <1> ; (INT_09h - Retro UNIX 8086 v1 - U9.ASM, 07/03/2014)
                        <1> ;
                        <1> ; Derived from "KB_INT_1" procedure of IBM "pc-at"
                        <1> ; rombios source code (06/10/1985)
                        <1> ; 'keybd.asm', HARDWARE INT 09h - (IRQ Level 1)
                        <1>
                        <1> ; EQUATES (IBM PC-XT-286 BIOS, 1986, 'POSQEQU.INC')
                        <1>
                        <1> ;--------- 8042 COMMANDS --------------------------------------------------------
                        <1> ENA_KBD             equ     0AEh            ; ENABLE KEYBOARD COMMAND
                        <1> DIS_KBD             equ     0ADh            ; DISABLE KEYBOARD COMMAND
                        <1> SHUT_CMD    equ     0FEh            ; CAUSE A SHUTDOWN COMMAND
                        <1> ;--------- 8042 KEYBOARD INTERFACE AND DIAGNOSTIC CONTROL REGISTERS ------------
                        <1> STATUS_PORT equ     064h            ; 8042 STATUS PORT
                        <1> INPT_BUF_FULL       equ     00000010b       ; 1 = +INPUT BUFFER FULL
                        <1> PORT_A              equ     060h            ; 8042 KEYBOARD SCAN CODE/CONTROL PORT
                        <1> ;---------- 8042 KEYBOARD RESPONSE ----------------------------------------------
                        <1> KB_ACK              equ     0FAh            ; ACKNOWLEDGE PROM TRANSMISSION
                        <1> KB_RESEND   equ     0FEh            ; RESEND REQUEST
                        <1> KB_OVER_RUN equ     0FFh            ; OVER RUN SCAN CODE
                        <1> ;---------- KEYBOARD/LED COMMANDS -----------------------------------------------
                        <1> KB_ENABLE   equ     0F4h            ; KEYBOARD ENABLE
                        <1> LED_CMD             equ     0EDh            ; LED WRITE COMMAND
                        <1> KB_TYPA_RD  equ     0F3h            ; TYPAMATIC RATE/DELAY COMMAND
                        <1> ;---------- KEYBOARD SCAN CODES -------------------------------------------------
```

```
                        <1> NUM_KEY             equ     69              ; SCAN CODE FOR         NUMBER LOCK KEY
                        <1> SCROLL_KEY  equ     70              ; SCAN CODE FOR         SCROLL LOCK KEY
                        <1> ALT_KEY             equ     56              ; SCAN CODE FOR         ALTERNATE SHIFT KEY
                        <1> CTL_KEY             equ     29              ; SCAN CODE FOR         CONTROL KEY
                        <1> CAPS_KEY    equ     58              ; SCAN CODE FOR         SHIFT LOCK KEY
                        <1> DEL_KEY             equ     83              ; SCAN CODE FOR         DELETE KEY
                        <1> INS_KEY             equ     82              ; SCAN CODE FOR         INSERT KEY
                        <1> LEFT_KEY    equ     42              ; SCAN CODE FOR         LEFT SHIFT
                        <1> RIGHT_KEY   equ     54              ; SCAN CODE FOR         RIGHT SHIFT
                        <1> SYS_KEY             equ     84              ; SCAN CODE FOR         SYSTEM KEY
                        <1> ;---------- ENHANCED KEYBOARD SCAN CODES ------------------------------------
                        <1> ID_1        equ     0ABh            ; 1ST ID CHARACTER FOR KBX
                        <1> ID_2        equ     041h            ; 2ND ID CHARACTER FOR KBX
                        <1> ID_2A               equ     054h            ; ALTERNATE 2ND ID CHARACTER FOR KBX
                        <1> F11_M               equ     87              ; F11 KEY MAKE
                        <1> F12_M               equ     88              ; F12 KEY MAKE
                        <1> MC_E0               equ     224             ; GENERAL MARKER CODE
                        <1> MC_E1               equ     225             ; PAUSE KEY MARKER CODE
                        <1> ;---------- FLAG EQUATES WITHIN @KB_FLAG-----------------------------------
                        <1> RIGHT_SHIFT equ     00000001b       ; RIGHT SHIFT KEY DEPRESSED
                        <1> LEFT_SHIFT  equ     00000010b       ; LEFT SHIFT KEY DEPRESSED
                        <1> CTL_SHIFT   equ     00000100b       ; CONTROL SHIFT KEY DEPRESSED
                        <1> ALT_SHIFT   equ     00001000b       ; ALTERNATE SHIFT KEY DEPRESSED
                        <1> SCROLL_STATE        equ     00010000b       ; SCROLL LOCK STATE IS ACTIVE
                        <1> NUM_STATE   equ     00100000b       ; NUM LOCK STATE IS ACTIVE
                        <1> CAPS_STATE  equ     01000000b       ; CAPS LOCK STATE IS ACTIVE
                        <1> INS_STATE   equ     10000000b       ; INSERT STATE IS ACTIVE
                        <1> ;---------- FLAG EQUATES WITHIN   @KB_FLAG_1 -------------------------------
                        <1> L_CTL_SHIFT equ     00000001b       ; LEFT CTL KEY DOWN
                        <1> L_ALT_SHIFT equ     00000010b       ; LEFT ALT KEY DOWN
                        <1> SYS_SHIFT   equ     00000100b       ; SYSTEM KEY DEPRESSED AND HELD
                        <1> HOLD_STATE  equ     00001000b       ; SUSPEND KEY HAS BEEN TOGGLED
                        <1> SCROLL_SHIFT        equ     00010000b       ; SCROLL LOCK KEY IS DEPRESSED
                        <1> NUM_SHIFT   equ     00100000b       ; NUM LOCK KEY IS DEPRESSED
                        <1> CAPS_SHIFT  equ     01000000b       ; CAPS LOCK KEY IS DEPRE55ED
                        <1> INS_SHIFT   equ     10000000b       ; INSERT KEY IS DEPRESSED
                        <1> ;---------- FLAGS EQUATES WITHIN @KB_FLAG_2 ---------------------------------
                        <1> KB_LEDS             equ     00000111b       ; KEYBOARD LED STATE BITS
                        <1> ;           equ     00000001b       ; SCROLL LOCK INDICATOR
                        <1> ;           equ     00000010b       ; NUM LOCK INDICATOR
                        <1> ;           equ     00000100b       ; CAPS LOCK INDICATOR
                        <1> ;           equ     00001000b       ; RESERVED (MUST BE ZERO)
                        <1> KB_FA               equ     00010000b       ; ACKNOWLEDGMENT RECEIVED
                        <1> KB_FE               equ     00100000b       ; RESEND RECEIVED FLAG
                        <1> KB_PR_LED   equ     01000000b       ; MODE INDICATOR UPDATE
                        <1> KB_ERR              equ     10000000b       ; KEYBOARD TRANSMIT ERROR FLAG
                        <1> ;---------- FLAGS EQUATES WITHIN @KB_FLAG_3 ---------------------------------
                        <1> LC_E1               equ     00000001b       ; LAST CODE WAS THE E1 HIDDEN CODE
                        <1> LC_E0               equ     00000010b       ; LAST CODE WAS THE E0 HIDDEN CODE
                        <1> R_CTL_SHIFT equ     00000100b       ; RIGHT CTL KEY DOWN
                        <1> R_ALT_SHIFT equ     00001000b       ; RIGHT ALT KEY DOWN
                        <1> GRAPH_ON    equ     00001000b       ; ALT GRAPHICS KEY DOWN (WT ONLY)
                        <1> KBX         equ     00010000b       ; ENHANCED KEYBOARD INSTALLED
                        <1> SET_NUM_LK  equ     00100000b       ; FORCE NUM LOCK IF READ ID AND KBX
                        <1> LC_AB               equ     01000000b       ; LAST CHARACTER WAS FIRST ID CHARACTER
                        <1> RD_ID               equ     10000000b       ; DOING A READ ID (MUST BE BIT0)
                        <1> ;
                        <1> ;----------- INTERRUPT EQUATES -------------------------------------------------
                        <1> EOI         equ     020h            ; END OF INTERRUPT COMMAND TO 8259
                        <1> INTA00              equ     020h            ; 8259 PORT
                        <1>
                        <1>
                        <1> kb_int:
                        <1>
                        <1> ; 24/07/2022 - TRDOS 386 v2.0.5
                        <1> ; 12/04/2021 - TRDOS 386 v2.0.3 (32 bit push/pop)
                        <1> ; 17/10/2015 ('ctrlbrk')
                        <1> ; 05/12/2014
                        <1> ; 04/12/2014 (derived from pc-xt-286 bios source code -1986-)
                        <1> ; 26/08/2014
                        <1> ;
                        <1> ; 03/06/86  KEYBOARD BIOS
                        <1> ;
                        <1> ;--- HARDWARE INT 09H -- (IRQ LEVEL 1) ----------------------------------------
                        <1> ;                                                                           ;
                        <1> ;   KEYBOARD INTERRUPT ROUTINE                                              ;
                        <1> ;                                                                           ;
                        <1> ;-------------------------------------------------------------------------------
                        <1>
                        <1> KB_INT_1:
000010ED FB             <1>     sti                             ; ENABLE INTERRUPTS
                        <1>     ;push   ebp
000010EE 50             <1>     push    eax
000010EF 53             <1>     push    ebx
000010F0 51             <1>     push    ecx
000010F1 52             <1>     push    edx
000010F2 56             <1>     push    esi
000010F3 57             <1>     push    edi
000010F4 1E             <1>     push    ds
000010F5 06             <1>     push    es
000010F6 FC             <1>     cld                             ; FORWARD DIRECTION
000010F7 66B81000       <1>     mov     ax, KDATA
000010FB 8ED8           <1>     mov     ds, ax
000010FD 8EC0           <1>     mov     es, ax
                        <1>     ;
                        <1>     ;-----  WAIT FOR KEYBOARD DISABLE COMMAND TO BE ACCEPTED
000010FF B0AD           <1>     mov     al, DIS_KBD             ; DISABLE THE KEYBOARD COMMAND
00001101 E84B050000     <1>     call    SHIP_IT                 ; EXECUTE DISABLE
00001106 FA             <1>     cli                             ; DISABLE INTERRUPTS
00001107 B900000100     <1>     mov     ecx, 10000h             ; SET MAXIMUM TIMEOUT
                        <1> KB_INT_01:
0000110C E464           <1>     in      al, STATUS_PORT                 ; READ ADAPTER STATUS
0000110E A802           <1>     test    al, INPT_BUF_FULL       ; CHECK INPUT BUFFER FULL STATUS BIT
00001110 E0FA           <1>     loopnz  KB_INT_01               ; WAIT FOR COMMAND TO BE ACCEPTED
                        <1>     ;
                        <1>     ;-----  READ CHARACTER FROM KEYBOARD INTERFACE
00001112 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
00001114 FB             <1>     sti                             ; ENABLE INTERRUPTS AGAIN
00001115 3CFE           <1>     cmp     al, KB_RESEND           ; IS THE INPUT A RESEND
00001117 740E           <1>         je      short KB_INT_4          ; GO IF RESEND
                        <1>     ;
                        <1>     ;-----  CHECK FOR RESPONSE TO A COMMAND TO KEYBOARD
```

```
00001119 3CFA                <1>     cmp     al, KB_ACK              ; IS THE INPUT AN ACKNOWLEDGE
0000111B 7514                <1>         jne     short KB_INT_2          ; GO IF NOT
                             <1>     ;
                             <1>     ;----- A COMMAND TO THE KEYBOARD WAS ISSUED
0000111D FA                  <1>     cli                             ; DISABLE INTERRUPTS
0000111E 800D[EB670000]10    <1>     or      byte [KB_FLAG_2], KB_FA ; INDICATE ACK RECEIVED
                             <1>         ;jmp       K26                  ; RETURN IF NOT ACK RETURNED FOR DATA)
                             <1>     ; 12/04/2021
00001125 EB76                <1>     jmp     short ID_EX  ; K26
                             <1>     ;
                             <1>     ;----- RESEND THE LAST BYTE
                             <1> KB_INT_4:
00001127 FA                  <1>     cli                             ; DISABLE INTERRUPTS
00001128 800D[EB670000]20    <1>     or      byte [KB_FLAG_2], KB_FE ; INDICATE RESEND RECEIVED
                             <1>         ;jmp       K26                  ; RETURN IF NOT ACK RETURNED FOR DATA)
                             <1>     ; 12/04/2021
0000112F EB6C                <1>     jmp     short ID_EX  ; K26
                             <1>     ;
                             <1> ;-----      UPDATE MODE INDICATORS IF CHANGE IN STATE
                             <1> KB_INT_2:
                             <1>     ;push   ax                      ; SAVE DATA IN
                             <1>     ; 12/04/2021
00001131 50                  <1>     push    eax
00001132 E8E0050000          <1>     call    MAKE_LED                ; GO GET MODE INDICATOR DATA BYTE
00001137 8A1D[EB670000]      <1>     mov     bl, [KB_FLAG_2]         ; GET PREVIOUS BITS
0000113D 30C3                <1>     xor     bl, al                  ; SEE IF ANY DIFFERENT
0000113F 80E307              <1>     and     bl, KB_LEDS             ; ISOLATE INDICATOR BITS
00001142 7405                <1>     jz      short UP0               ; IF NO CHANGE BYPASS UPDATE
00001144 E863050000          <1>     call    SND_LED                 ; GO TURN ON MODE INDICATORS
                             <1> UP0:
                             <1>     ;pop    ax                      ; RESTORE DATA IN
                             <1>     ; 12/04/2021
00001149 58                  <1>     pop     eax
                             <1> ;--------------------------------------------------------------------------
                             <1> ;   START OF KEY PROCESSING                                             ;
                             <1> ;--------------------------------------------------------------------------
0000114A 88C4                <1>     mov     ah, al                  ; SAVE SCAN CODE IN AH ALSO
                             <1>     ;
                             <1>     ;----- TEST FOR OVERRUN SCAN CODE FROM KEYBOARD
0000114C 3CFF                <1>     cmp     al, KB_OVER_RUN                 ; IS THIS AN OVERRUN CHAR
                             <1>         ;je K62                     ; BUFFER_FULL_BEEP
                             <1>     ; 12/04/2021
0000114E 7505                <1>     jne     short K16
00001150 E9E8040000          <1>     jmp     K62
                             <1> K16:
00001155 8A3D[EC670000]      <1>     mov     bh, [KB_FLAG_3]                 ; LOAD FLAGS FOR TESTING
                             <1>     ;
                             <1>     ;----- TEST TO SEE IF A READ_ID IS IN PROGRESS
0000115B F6C7C0              <1>     test    bh, RD_ID+LC_AB         ; ARE WE DOING A READ ID?
0000115E 7442                <1>     jz      short NOT_ID            ; CONTINUE IF NOT
00001160 7914                <1>     jns     short TST_ID_2          ; IS THE RD_ID FLAG ON?
00001162 3CAB                <1>     cmp     al, ID_1                ; IS THIS THE 1ST ID CHARACTER?
00001164 7507                <1>     jne     short RST_RD_ID
00001166 800D[EC670000]40    <1>     or      byte [KB_FLAG_3], LC_AB ; INDICATE 1ST ID WAS OK
                             <1> RST_RD_ID:
0000116D 8025[EC670000]7F    <1>     and     byte [KB_FLAG_3], ~RD_ID ; RESET THE READ ID FLAG
00001174 EB27                <1>         jmp short ID_EX             ; AND EXIT
                             <1>     ; 12/04/2021
                             <1>     ;jmp    K26
                             <1>     ;
                             <1> TST_ID_2:
00001176 8025[EC670000]BF    <1>     and     byte [KB_FLAG_3], ~LC_AB ; RESET FLAG
0000117D 3C54                <1>     cmp     al, ID_2A               ; IS THIS THE 2ND ID CHARACTER?
0000117F 7415                <1>         je  short KX_BIT            ; JUMP IF SO
00001181 3C41                <1>     cmp     al, ID_2                ; IS THIS THE 2ND ID CHARACTER?
00001183 7518                <1>         jne short ID_EX             ; LEAVE IF NOT
                             <1>     ; 12/04/2021
                             <1>     ;jne    K26
                             <1>     ;
                             <1>     ;----- A READ ID SAID THAT IT WAS ENHANCED KEYBOARD
00001185 F6C720              <1>     test    bh, SET_NUM_LK                  ; SHOULD WE SET NUM LOCK?
00001188 740C                <1>         jz      short KX_BIT                ; EXIT IF NOT
0000118A 800D[E9670000]20    <1>     or      byte [KB_FLAG], NUM_STATE ; FORCE NUM LOCK ON
00001191 E816050000          <1>     call    SND_LED                 ; GO SET THE NUM LOCK INDICATOR
                             <1> KX_BIT:
00001196 800D[EC670000]10    <1>     or      byte [KB_FLAG_3], KBX ; INDICATE ENHANCED KEYBOARD WAS FOUND
0000119D E9CB010000          <1> ID_EX:      jmp     K26                     ; EXIT
                             <1>     ;
                             <1> NOT_ID:
000011A2 3CE0                <1>     cmp     al, MC_E0               ; IS THIS THE GENERAL MARKER CODE?
000011A4 7509                <1>     jne     short TEST_E1
000011A6 800D[EC670000]12    <1>     or      byte [KB_FLAG_3], LC_E0+KBX ; SET FLAG BIT, SET KBX, AND
000011AD EB0B                <1>     jmp     short EXIT              ; THROW AWAY THIS CODE
                             <1>     ; 12/04/2021
                             <1>     ;jmp    K26A
                             <1> TEST_E1:
000011AF 3CE1                <1>     cmp     al, MC_E1               ; IS THIS THE PAUSE KEY?
000011B1 750C                <1>     jne     short NOT_HC
000011B3 800D[EC670000]11    <1>     or      byte [KB_FLAG_3], LC_E1+KBX ; SET FLAG BIT, SET KBX, AND
000011BA E9B5010000          <1> EXIT:       jmp     K26A                    ; THROW AWAY THIS CODE
                             <1>     ;
                             <1> NOT_HC:
000011BF 247F                <1>     and     al, 07Fh                ; TURN OFF THE BREAK BIT
000011C1 F6C702              <1>     test    bh, LC_E0               ; LAST CODE THE E0 MARKER CODE
000011C4 7410                <1>     jz      short NOT_LC_E0                 ; JUMP IF NOT
                             <1>     ;
000011C6 BF[D6660000]        <1>     mov     edi, _K6+6              ; IS THIS A SHIFT KEY?
000011CB AE                  <1>     scasb
                             <1>     ;je     K26 ; K16B                ; YES, THROW AWAY & RESET FLAG
                             <1>     ; 12/04/2021
000011CC 745B                <1>     je      short K16B ; K26
000011CE AE                  <1>     scasb
000011CF 756D                <1>     jne     short K16A              ; NO, CONTINUE KEY PROCESSING
                             <1>     ;jmp    short K16B              ; YES, THROW AWAY & RESET FLAG
000011D1 E997010000          <1>     jmp     K26
                             <1>     ;
                             <1> NOT_LC_E0:
000011D6 F6C701              <1>     test    bh, LC_E1               ; LAST CODE THE E1 MARKER CODE?
000011D9 7425                <1>     jz      short T_SYS_KEY                 ; JUMP IF NOT
000011DB B904000000          <1>     mov     ecx, 4                  ; LENGHT OF SEARCH
000011E0 BF[D4660000]        <1>     mov     edi, _K6+4              ; IS THIS AN ALT, CTL, OR SHIFT?
000011E5 F2AE                <1>     repne   scasb                   ; CHECK IT
000011E7 74D1                <1>     je      short EXIT              ; THROW AWAY IF SO
                             <1>     ; 12/04/2021
                             <1>     ;je     K26A
                             <1>     ;
000011E9 3C45                <1>     cmp     al, NUM_KEY             ; IS IT THE PAUSE KEY?
000011EB 753C                <1>     jne     short K16B              ; NO, THROW AWAY & RESET FLAG
                             <1>     ; 12/04/2021
                             <1>     ;jne    K26
000011ED F6C480              <1>     test    ah, 80h                 ; YES, IS IT THE BREAK OF THE KEY?
000011F0 7537                <1>     jnz     short K16B              ; YES, THROW THIS AWAY, TOO
                             <1>     ; 24/07/2022
                             <1>     ;jnz    K26
```

```
                                <1>         ; 20/02/2015
000011F2 F605[EA670000]08       <1>     test   byte [KB_FLAG_1],HOLD_STATE ; NO, ARE WE PAUSED ALREADY?
000011F9 752E                   <1>     jnz    short K16B             ; YES, THROW AWAY
                                <1>     ; 12/04/2021
                                <1>     ;jnz   K26
000011FB E9B1020000             <1>     jmp     K39P                    ; NO, THIS IS THE REAL PAUSE STATE
                                <1>     ;
                                <1>     ;----- TEST FOR SYSTEM KEY
                                <1> T_SYS_KEY:
00001200 3C54                   <1>     cmp    al, SYS_KEY             ; IS IT THE SYSTEM KEY?
00001202 753A                   <1>     jnz    short K16A              ; CONTINUE IF NOT
                                <1>     ;
00001204 F6C480                 <1>     test   ah, 80h                 ; CHECK IF THIS A BREAK CODE
00001207 7525                   <1>     jnz    short K16C              ; DO NOT TOUCH SYSTEM INDICATOR IF TRUE
                                <1>     ;
00001209 F605[EA670000]04       <1>     test   byte [KB_FLAG_1], SYS_SHIFT ; SEE IF IN SYSTEM KEY HELD DOWN
00001210 7517                   <1>     jnz    short K16B              ; IF YES, DO NOT PROCESS SYSTEM INDICATOR
                                <1>     ; 12/04/2021
                                <1>     ;jnz   K26
                                <1>     ;
00001212 800D[EA670000]04       <1>     or     byte [KB_FLAG_1], SYS_SHIFT ; INDICATE SYSTEM KEY DEPRESSED
00001219 B020                   <1>     mov    al, EOI                 ; END OF INTERRUPT COMMAND
0000121B E620                   <1>     out    20h, al ;out INTA00, al       ; SEND COMMAND TO INTERRUPT CONTROL PORT
                                <1>                                    ; INTERRUPT-RETURN-NO-EOI
0000121D B0AE                   <1>     mov    al, ENA_KBD             ; INSURE KEYBOARD IS ENABLED
0000121F E82D040000             <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
00001224 E957010000             <1>        jmp     K27A                    ; END PROCESSING
                                <1>     ;
00001229 E93F010000             <1> K16B:       jmp    K26                  ; IGNORE SYSTEM KEY
                                <1>     ;
                                <1> K16C:
0000122E 8025[EA670000]FB       <1>     and    byte [KB_FLAG_1], ~SYS_SHIFT ; TURN OFF SHIFT KEY HELD DOWN
00001235 B020                   <1>     mov    al, EOI                 ; END OF INTERRUPT COMMAND
00001237 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>     ;
00001239 E93B010000             <1>     jmp     K27                    ; IGNORE SYSTEM KEY
                                <1>     ;
                                <1>     ;----- TEST FOR SHIFT KEYS
                                <1> K16A:
0000123E 8A1D[E9670000]         <1>     mov    bl, [KB_FLAG]           ; PUT STATE FLAGS IN BL
00001244 BF[D0660000]           <1>     mov    edi, _K6                ; SHIFT KEY TABLE offset
00001249 B908000000             <1>     mov    ecx, _K6L               ; LENGTH
0000124E F2AE                   <1>     repne  scasb                   ; LOOK THROUGH THE TABLE FOR A MATCH
00001250 88E0                   <1>     mov    al, ah                  ; RECOVER SCAN CODE
                                <1>        ;jne       K25                     ; IF NO MATCH, THEN SHIFT NOT FOUND
                                <1>     ; 12/04/2021
00001252 7405                   <1>     je     short K17
00001254 E9FC000000             <1>     jmp    K25
                                <1>     ;
                                <1>     ;------SHIFT KEY FOUND
                                <1> K17:
00001259 81EF[D1660000]         <1>        sub     edi, _K6+1              ; ADJUST PTR TO SCAN CODE MATCH
0000125F 8AA7[D8660000]         <1>           mov     ah, [edi+_K7]         ; GET MASK INTO AH
00001265 B102                   <1>     mov    cl, 2                   ; SETUP COUNT FOR FLAG SHIFTS
00001267 A880                   <1>     test   al, 80h                 ; TEST FOR BREAK KEY
                                <1>     ;jnz   short K23               ; JUMP OF BREAK
                                <1>     ; 12/04/2021
00001269 7405                   <1>     jz     short K17C
0000126B E981000000             <1>     jmp    K23
                                <1>     ;
                                <1>     ;-----  SHIFT MAKE FOUND, DETERMINE SET OR TOGGLE
                                <1> K17C:
00001270 80FC10                 <1>     cmp    ah, SCROLL_SHIFT
00001273 7324                   <1>     jae    short K18               ; IF SCROLL SHIFT OR ABOVE, TOGGLE KEY
                                <1>     ;
                                <1>     ;-----  PLAIN SHIFT KEY, SET SHIFT ON
00001275 0825[E9670000]         <1>     or     [KB_FLAG], ah           ; TURN ON SHIFT BIT
0000127B A80C                   <1>        test       al, CTL_SHIFT+ALT_SHIFT ; IS IT ALT OR CTRL?
                                <1>     ;;jnz  short K17D              ; YES, MORE FLAGS TO SET
                                <1>     ;jz    K26                     ; NO, INTERRUPT RETURN
                                <1>     ; 12/04/2021
0000127D 7415                   <1>     jz     short k17f
                                <1> K17D:
0000127F F6C702                 <1>     test   bh, LC_E0               ; IS THIS ONE OF NEW KEYS?
00001282 7408                   <1>     jz     short K17E              ; NO, JUMP
00001284 0825[EC670000]         <1>     or     [KB_FLAG_3], ah                ; SET BITS FOR RIGHT CTRL, ALT
                                <1>     ;jmp   K26                     ; INTERRUPT RETURN
                                <1>     ; 12/04/2021
0000128A EB08                   <1>     jmp    short k17f
                                <1> K17E:
0000128C D2EC                   <1>     shr    ah, cl                  ; MOVE FLAG BITS TWO POSITIONS
0000128E 0825[EA670000]         <1>     or     [KB_FLAG_1], ah                ; SET BITS FOR LEFT CTRL, ALT
                                <1> k17f:       ; 12/04/2021
00001294 E9D4000000             <1>     jmp    K26
                                <1>     ;
                                <1>     ;-----  TOGGLED SHIFT KEY, TEST FOR 1ST MAKE OR NOT
                                <1> K18:                               ; SHIFT-TOGGLE
00001299 F6C304                 <1>     test   bl, CTL_SHIFT           ; CHECK CTL SHIFT STATE
0000129C 7402                   <1>     jz     short K18A                ; JUMP IF NOT CTL STATE
                                <1>        ;jnz       K25                     ; JUMP IF CTL STATE
                                <1>     ; 12/04/2021
0000129E EB1C                   <1>     jmp    short k20a ; K25
                                <1> K18A:
000012A0 3C52                   <1>     cmp    al, INS_KEY             ; CHECK FOR INSERT KEY
000012A2 7522                   <1>     jne    short K22               ; JUMP IF NOT INSERT KEY
000012A4 F6C308                 <1>     test   bl, ALT_SHIFT           ; CHECK FOR ALTERNATE SHIFT
000012A7 7402                   <1>           jz     short K18B              ; JUMP IF NOT ALTERNATE SHIFT
                                <1>        ;jnz       K25                     ; JUMP IF ALTERNATE SHIFT
                                <1>     ; 12/04/2021
000012A9 EB11                   <1>     jmp    short k20a ; K25
                                <1> K18B:
000012AB F6C702                 <1>     test   bh, LC_E0 ;20/02/2015 ; IS THIS NEW INSERT KEY?
000012AE 7516                   <1>     jnz    short K22               ; YES, THIS ONE'S NEVER A '0'
                                <1> K19:
000012B0 F6C320                 <1>     test   bl, NUM_STATE           ; CHECK FOR BASE STATE
000012B3 750C                   <1>     jnz    short K21               ; JUMP IF NUM LOCK IS ON
000012B5 F6C303                 <1>     test   bl, LEFT_SHIFT+RIGHT_SHIFT ; TEST FOR SHIFT STATE
000012B8 740C                   <1>     jz     short K22               ; JUMP IF BASE STATE
                                <1> K20:                               ; NUMERIC ZERO, NOT INSERT KEY
000012BA 88C4                   <1>     mov    ah, al                  ; PUT SCAN CODE BACK IN AH
                                <1> k20a:       ; 12/04/2021
000012BC E994000000             <1>         jmp     K25                    ; NUMERAL '0', STNDRD. PROCESSING
```

```
                        <1> K21:                                ; MIGHT BE NUMERIC
000012C1 F6C303         <1>     test    bl, LEFT_SHIFT+RIGHT_SHIFT
000012C4 74F4           <1>     jz      short K20               ; IS NUMERIC, STD. PROC.
                        <1>     ;
                        <1> K22:                                ; SHIFT TOGGLE KEY HIT; PROCESS IT
000012C6 8425[EA670000] <1>     test    ah, [KB_FLAG_1]         ; IS KEY ALREADY DEPRESSED
                        <1>         ;jnz        short K26               ; JUMP IF KEY ALREADY DEPRESSED
                        <1>     ; 12/04/2021
000012CC 75C6           <1>     jnz     short k17f ; K26
                        <1> K22A:
000012CE 0825[EA670000] <1>         or      [KB_FLAG_1], ah     ; INDICATE THAT THE KEY IS DEPRESSED
000012D4 3025[E9670000] <1>     xor     [KB_FLAG], ah           ; TOGGLE THE SHIFT STATE
                        <1>     ;
                        <1>     ;-----  TOGGLE LED IF CAPS, NUM  OR SCROLL KEY DEPRESSED
000012DA F6C470         <1>     test    ah, CAPS_SHIFT+NUM_SHIFT+SCROLL_SHIFT ; SHIFT TOGGLE?
000012DD 7407           <1>     jz      short K22B              ; GO IF NOT
                        <1>     ;
                        <1>     ; 12/04/2021 (32 bit push/pop)
000012DF 50             <1>     push    eax ; push ax           ; SAVE SCAN CODE AND SHIFT MASK
000012E0 E8C7030000     <1>     call    SND_LED                 ; GO TURN MODE INDICATORS ON
000012E5 58             <1>     pop     eax ; pop ax            ; RESTORE SCAN CODE
                        <1> K22B:
000012E6 3C52           <1>     cmp     al, INS_KEY             ; TEST FOR 1ST MAKE OF INSERT KEY
                        <1>         ;jne        short K26               ; JUMP IF NOT INSERT KEY
                        <1>     ; 12/04/2021
000012E8 75AA           <1>     jne     short k17f ; K26
000012EA 88C4           <1>     mov     ah, al                   ; SCAN CODE IN BOTH HALVES OF AX
000012EC E999000000     <1>         jmp K28                      ; FLAGS UPDATED, PROC. FOR BUFFER
                        <1>     ;
                        <1>     ;-----  BREAK SHIFT FOUND
                        <1> K23:                                ; BREAK-SHIFT-FOUND
000012F1 80FC10         <1>     cmp     ah, SCROLL_SHIFT        ; IS THIS A TOGGLE KEY
000012F4 F6D4           <1>     not     ah                      ; INVERT MASK
000012F6 7355           <1>     jae     short K24               ; YES, HANDLE BREAK TOGGLE
000012F8 2025[E9670000] <1>     and     [KB_FLAG], ah           ; TURN OFF SHIFT BIT
000012FE 80FCFB         <1>     cmp     ah, ~CTL_SHIFT          ; IS THIS ALT OR CTL?
00001301 7730           <1>     ja      short K23D              ; NO, ALL DONE
                        <1>     ;
00001303 F6C702         <1>     test    bh, LC_E0               ; 2ND ALT OR CTL?
00001306 7408           <1>     jz      short K23A              ; NO, HANSLE NORMALLY
00001308 2025[EC670000] <1>     and     [KB_FLAG_3], ah             ; RESET BIT FOR RIGHT ALT OR CTL
0000130E EB08           <1>     jmp     short K23B              ; CONTINUE
                        <1> K23A:
00001310 D2FC           <1>     sar     ah, cl                  ; MOVE THE MASK BIT TWO POSITIONS
00001312 2025[EA670000] <1>     and     [KB_FLAG_1], ah             ; RESET BIT FOR LEFT ALT AND CTL
                        <1> K23B:
00001318 88C4           <1>     mov     ah, al                  ; SAVE SCAN CODE
0000131A A0[EC670000]   <1>     mov     al, [KB_FLAG_3]             ; GET RIGHT ALT & CTRL FLAGS
0000131F D2E8           <1>     shr     al, cl                  ; MOVE TO BITS 1 & 0
00001321 0A05[EA670000] <1>     or      al, [KB_FLAG_1]             ; PUT IN LEFT ALST & CTL FLAGS
00001327 D2E0           <1>     shl     al, cl                  ; MOVE BACK TO BITS 3 & 2
00001329 240C           <1>     and     al, ALT_SHIFT+CTL_SHIFT ; FILTER OUT OTHER GARBAGE
0000132B 0805[E9670000] <1>     or      [KB_FLAG], al           ; PUT RESULT IN THE REAL FLAGS
00001331 88E0           <1>     mov     al, ah
                        <1> K23D:
00001333 3CB8           <1>     cmp     al, ALT_KEY+80h             ; IS THIS ALTERNATE SHIFT RELEASE
00001335 7536           <1>     jne     short K26               ; INTERRUPT RETURN
                        <1>     ;
                        <1>     ;-----  ALTERNATE SHIFT KEY RELEASED, GET THE VALUE INTO BUFFER
00001337 A0[ED670000]   <1>     mov     al, [ALT_INPUT]
0000133C B400           <1>     mov     ah, 0                   ; SCAN CODE OF 0
0000133E 8825[ED670000] <1>     mov     [ALT_INPUT], ah         ; ZERO OUT THE FIELD
00001344 3C00           <1>     cmp     al, 0                   ; WAS THE INPUT = 0?
00001346 7425           <1>     je      short K26               ; INTERRUPT_RETURN
                        <1>         ; 29/01/2016
                        <1>     ;jmp    K61                     ; IT WASN'T, SO PUT IN BUFFER
00001348 E9AB020000     <1>     jmp     _K60
                        <1>     ;
                        <1> K24:                                ; BREAK-TOGGLE
0000134D 2025[EA670000] <1>     and     [KB_FLAG_1], ah         ; INDICATE NO LONGER DEPRESSED
00001353 EB18           <1>     jmp     short K26               ; INTERRUPT_RETURN
                        <1>     ;
                        <1>     ;-----  TEST FOR HOLD STATE
                        <1>                                     ; AL, AH = SCAN CODE
                        <1> K25:                                ; NO-SHIFT-FOUND
00001355 3C80           <1>     cmp     al, 80h                 ; TEST FOR BREAK KEY
00001357 7314           <1>     jae     short K26               ; NOTHING FOR BREAK CHARS FROM HERE ON
00001359 F605[EA670000]08   <1>     test    byte [KB_FLAG_1], HOLD_STATE ; ARE WE IN HOLD STATE
00001360 7428           <1>     jz      short K28               ; BRANCH AROUND TEST IF NOT
00001362 3C45           <1>     cmp     al, NUM_KEY
00001364 7407           <1>     je      short K26               ; CAN'T END HOLD ON NUM_LOCK
00001366 8025[EA670000]F7   <1>     and     byte [KB_FLAG_1], ~HOLD_STATE ; TURN OFF THE HOLD STATE BIT
                        <1> K26:
0000136D 8025[EC670000]FC   <1>     and     byte [KB_FLAG_3], ~(LC_E0+LC_E1) ; RESET LAST CHAR H.C. FLAG
                        <1> K26A:                                   ; INTERRUPT-RETURN
00001374 FA             <1>     cli                             ; TURN OFF INTERRUPTS
00001375 B020           <1>     mov     al, EOI                 ; END OF INTERRUPT COMMAND
00001377 E620           <1>     out     20h, al ;out INTA00, al         ; SEND COMMAND TO INTERRUPT CONTROL PORT
                        <1> K27:                                ; INTERRUPT-RETURN-NO-EOI
00001379 B0AE           <1>     mov     al, ENA_KBD             ; INSURE KEYBOARD IS ENABLED
0000137B E8D1020000     <1>     call    SHIP_IT                 ; EXECUTE ENABLE
                        <1> K27A:
00001380 FA             <1>     cli                             ; DISABLE INTERRUPTS
                        <1>     ;;mov   byte [intflg], 0 ; 07/01/2017 ;; 15/01/2017
00001381 07             <1>     pop     es                      ; RESTORE REGISTERS
00001382 1F             <1>     pop     ds
00001383 5F             <1>     pop     edi
00001384 5E             <1>     pop     esi
00001385 5A             <1>     pop     edx
00001386 59             <1>     pop     ecx
00001387 5B             <1>     pop     ebx
00001388 58             <1>     pop     eax
                        <1>     ;pop    ebp
00001389 CF             <1>     iretd                           ; RETURN
                        <1>
                        <1>     ;-----  NOT IN  HOLD STATE
                        <1> K28:                                ; NO-HOLD-STATE
0000138A 3C58           <1>     cmp     al, 88                  ; TEST FOR OUT-OF-RANGE SCAN CODES
0000138C 77DF           <1>     ja      short K26               ; IGNORE IF OUT-OF-RANGE
                        <1>     ;
0000138E F6C308         <1>     test    bl, ALT_SHIFT           ; ARE WE IN ALTERNATE SHIFT
00001391 740E           <1>         jz  short K28A              ; IF NOT ALTERNATE
                        <1>         ; 12/04/2021
                        <1>     ;jz     K38
                        <1>     ;
00001393 F6C710         <1>     test    bh, KBX                 ; IS THIS THE ENCHANCED KEYBOARD?
00001396 740E           <1>     jz      short K29               ; NO, ALT STATE IS REAL
                        <1>      ;28/02/2015
00001398 F605[EA670000]04   <1>     test    byte [KB_FLAG_1], SYS_SHIFT ; YES, IS SYSREQ KEY DOWN?
0000139F 7405           <1>     jz      short K29               ;  NO, ALT STATE IS REAL
                        <1>     ; 12/04/2021
                        <1>     ;jnz    K38                     ; YES, THIS IS PHONY ALT STATE
                        <1>         ;                           ; DUE TO PRESSING SYSREQ
000013A1 E9C4000000     <1> K28A:       jmp     K38
```

```
 997                                    <1>     ;
 998                                    <1>     ;----- TEST FOR RESET KEY SEQUENCE (CTL ALT DEL)
 999                                    <1> K29:                                ; TEST-RESET
1000 000013A6 F6C304                    <1>     test    bl, CTL_SHIFT           ; ARE WE IN CONTROL SHIFT ALSO?
1001 000013A9 740B                      <1>     jz      short K31               ; NO_RESET
1002 000013AB 3C53                      <1>     cmp     al, DEL_KEY             ; CTL-ALT STATE, TEST FOR DELETE KEY
1003 000013AD 7507                      <1>     jne     short K31               ; NO_RESET, IGNORE
1004                                    <1>     ;
1005                                    <1>     ;----- CTL-ALT-DEL HAS BEEN FOUND
1006                                    <1>     ; 26/08/2014
1007                                    <1> cpu_reset:
1008                                    <1>     ; IBM PC/AT ROM BIOS source code - 10/06/85 (TEST4.ASM - PROC_SHUTDOWN)
1009                                    <1>     ; Send FEh (system reset command) to the keyboard controller.
1010 000013AF B0FE                      <1>     mov     al, SHUT_CMD            ; SHUTDOWN COMMAND
1011 000013B1 E664                      <1>     out     STATUS_PORT, al                 ; SEND TO KEYBOARD CONTROL PORT
1012                                    <1> khere:
1013 000013B3 F4                        <1>     hlt                             ; WAIT FOR 80286 RESET
1014 000013B4 EBFD                      <1>     jmp     short khere             ; INSURE HALT
1015                                    <1>     ;
1016                                    <1>     ;-----  IN ALTERNATE SHIFT, RESET NOT FOUND
1017                                    <1> K31:                                ; NO-RESET
1018 000013B6 3C39                      <1>     cmp     al, 57                  ; TEST FOR SPACE KEY
1019 000013B8 7507                      <1>     jne     short K311              ; NOT THERE
1020 000013BA B020                      <1>     mov     al, ' '                 ; SET SPACE CHAR
1021                                    <1> k31a:       ; 12/04/2021
1022 000013BC E929020000                <1>         jmp     K57                     ; BUFFER_FILL
1023                                    <1> K311:
1024 000013C1 3C0F                      <1>     cmp     al, 15                  ; TEST FOR TAB KEY
1025 000013C3 7506                      <1>     jne     short K312              ; NOT THERE
1026 000013C5 66B800A5                  <1>     mov     ax, 0A500h              ; SET SPECIAL CODE FOR ALT-TAB
1027                                    <1>         ;jmp       K57                  ; BUFFER_FILL
1028                                    <1>     ; 12/04/2021
1029 000013C9 EBF1                      <1>     jmp     short k31a
1030                                    <1> K312:
1031 000013CB 3C4A                      <1>     cmp     al, 74                  ; TEST FOR KEY PAD -
1032                                    <1>         ;je short K37B              ; GO PROCESS
1033                                    <1>     ; 12/04/2021
1034 000013CD 7404                      <1>     je      short k312a
1035 000013CF 3C4E                      <1>     cmp     al, 78                  ; TEST FOR KEY PAD +
1036                                    <1>         ;je short K37B              ; GO PROCESS
1037                                    <1>     ; 12/04/2021
1038 000013D1 7505                      <1>     jne     short K32
1039                                    <1> k312a:
1040 000013D3 E988000000                <1>     jmp     K37B
1041                                    <1>     ;
1042                                    <1>     ;-----  LOOK FOR KEY PAD ENTRY
1043                                    <1> K32:                                ; ALT-KEY-PAD
1044 000013D8 BF[AC660000]              <1>     mov     edi, K30                ; ALT-INPUT-TABLE offset
1045 000013DD B90A000000                <1>     mov     ecx, 10                 ; LOOK FOR ENTRY USING KEYPAD
1046 000013E2 F2AE                      <1>     repne   scasb                   ; LOOK FOR MATCH
1047 000013E4 7521                      <1>     jne     short K33               ; NO_ALT_KEYPAD
1048 000013E6 F6C702                    <1>     test    bh, LC_E0               ; IS THIS ONE OF THE NEW KEYS?
1049 000013E9 7579                      <1>         jnz short K37C              ; YES, JUMP, NOT NUMPAD KEY
1050 000013EB 81EF[AD660000]            <1>     sub     edi, K30+1              ; DI NOW HAS ENTRY VALUE
1051 000013F1 A0[ED670000]              <1>     mov     al, [ALT_INPUT]         ; GET THE CURRENT BYTE
1052 000013F6 B40A                      <1>     mov     ah, 10                  ; MULTIPLY BY 10
1053 000013F8 F6E4                      <1>     mul     ah
1054 000013FA 6601F8                    <1>     add     ax, di                  ; ADD IN THE LATEST ENTRY
1055 000013FD A2[ED670000]              <1>     mov     [ALT_INPUT], al         ; STORE IT AWAY
1056                                    <1> K32A:
1057 00001402 E966FFFFFF                <1>         jmp     K26                     ; THROW AWAY THAT KEYSTROKE
1058                                    <1>     ;
1059                                    <1>     ;-----  LOOK FOR SUPERSHIFT ENTRY
1060                                    <1> K33:                                ; NO-ALT-KEYPAD
1061 00001407 C605[ED670000]00          <1>         mov     byte [ALT_INPUT], 0     ; ZERO ANY PREVIOUS ENTRY INTO INPUT
1062 0000140E B91A000000                <1>     mov     ecx, 26                 ; (DI),(ES) ALREADY POINTING
1063 00001413 F2AE                      <1>     repne   scasb                   ; LOOK FOR MATCH IN ALPHABET
1064 00001415 7445                      <1>     je      short K37A              ; MATCH FOUND, GO FILLL THE BUFFER
1065                                    <1>     ;
1066                                    <1>     ;-----  LOOK FOR TOP ROW OF ALTERNATE SHIFT
1067                                    <1> K34:                                ; ALT-TOP-ROW
1068 00001417 3C02                      <1>     cmp     al, 2                   ; KEY WITH '1' ON IT
1069 00001419 7245                      <1>     jb      short K37B              ; MUST BE ESCAPE
1070 0000141B 3C0D                      <1>     cmp     al, 13                  ; IS IT IN THE REGION
1071 0000141D 7705                      <1>     ja      short K35               ; NO, ALT SOMETHING ELSE
1072 0000141F 80C476                    <1>     add     ah, 118                 ; CONVERT PSEUDO SCAN CODE TO RANGE
1073 00001422 EB38                      <1>     jmp     short K37A              ; GO FILL THE BUFFER
1074                                    <1>     ;
1075                                    <1>     ;-----  TRANSLATE ALTERNATE SHIFT PSEUDO SCAN CODES
1076                                    <1> K35:                                ; ALT-FUNCTION
1077 00001424 3C57                      <1>     cmp     al, F11_M               ; IS IT F11?
1078 00001426 7209                      <1>     jb      short K35A ; 20/02/2015         ; NO, BRANCH
1079 00001428 3C58                      <1>     cmp     al, F12_M               ; IS IT F12?
1080 0000142A 7705                      <1>     ja      short K35A ; 20/02/2015         ; NO, BRANCH
1081 0000142C 80C434                    <1>     add     ah, 52                  ; CONVERT TO PSEUDO SCAN CODE
1082 0000142F EB2B                      <1>     jmp     short K37A              ; GO FILL THE BUFFER
1083                                    <1> K35A:
1084 00001431 F6C702                    <1>     test    bh, LC_E0               ; DO WE HAVE ONE OF THE NEW KEYS?
1085 00001434 741B                      <1>     jz      short K37               ; NO, JUMP
1086 00001436 3C1C                      <1>     cmp     al, 28                  ; TEST FOR KEYPAD ENTER
1087 00001438 7506                      <1>         jne     short K35B                  ; NOT THERE
1088 0000143A 66B800A6                  <1>     mov     ax, 0A600h              ; SPECIAL CODE
1089                                    <1>     ;jmp    K57                     ; BUFFER FILL
1090                                    <1>     ; 12/04/2021
1091 0000143E EB0C                      <1>     jmp     short k35c
1092                                    <1> K35B:
1093 00001440 3C53                      <1>     cmp     al, 83                  ; TEST FOR DELETE KEY
1094 00001442 7420                      <1>     je      short K37C              ; HANDLE WITH OTHER EDIT KEYS
1095 00001444 3C35                      <1>     cmp     al, 53                  ; TEST FOR KEYPAD /
1096 00001446 75BA                      <1>     jne     short K32A              ; NOT THERE, NO OTHER E0 SPECIALS
1097                                    <1>         ; 12/04/2021
1098                                    <1>     ;jne    K26
1099 00001448 66B800A4                  <1>     mov     ax, 0A400h              ; SPECIAL CODE1
1100                                    <1> k35c:       ; 12/04/2021
1101 0000144C E999010000                <1>     jmp     K57                     ; BUFFER FILL
1102                                    <1> K37:
1103 00001451 3C3B                      <1>     cmp     al, 59                  ; TEST FOR FUNCTION KEYS (F1)
1104 00001453 720B                      <1>         jb      short K37B          ; NO FN, HANDLE W/OTHER EXTENDED
1105 00001455 3C44                      <1>     cmp     al, 68                  ; IN KEYPAD REGION?
1106 00001457 77A9                      <1>     ja      short K32A              ; IF SO, IGNORE
1107                                    <1>     ; 12/04/2021
1108                                    <1>     ;ja     K26
1109 00001459 80C42D                    <1>     add     ah, 45                  ; CONVERT TO PSEUDO SCAN CODE
1110                                    <1> K37A:
1111 0000145C B000                      <1>     mov     al, 0                   ; ASCII CODE OF ZERO
1112                                    <1>     ;jmp    K57                     ; PUT IT IN THE BUFFER
1113                                    <1>     ; 12/04/2021
1114 0000145E EBEC                      <1>     jmp     short k35c
1115                                    <1> K37B:
1116 00001460 B0F0                      <1>     mov     al, 0F0h                ; USE SPECIAL ASCII CODE
1117                                    <1>     ;jmp    K57                     ; PUT IT IN THE BUFFER
1118                                    <1>     ; 12/04/2021
1119 00001462 EBE8                      <1>     jmp     short k35c
1120                                    <1> K37C:
```

```
1121 00001464 0450                <1>     add     al, 80                  ; CONVERT SCAN CODE (EDIT KEYS)
1122 00001466 88C4                <1>     mov     ah, al                  ; (SCAN CODE NOT IN AH FOR INSERT)
1123 00001468 EBF2                <1>     jmp     short K37A                ; PUT IT IN THE BUFFER
1124                              <1>     ;
1125                              <1>     ;----- NOT IN ALTERNATE SHIFT
1126                              <1> K38:                                ; NOT-ALT-SHIFT
1127                              <1>                                     ; BL STILL HAS SHIFT FLAGS
1128 0000146A F6C304              <1>     test    bl, CTL_SHIFT           ; ARE WE IN CONTROL SHIFT?
1129                              <1>     ;;jnz   short K38A              ; YES, START PROCESSING
1130                              <1>         ;jz K44                     ; NOT-CTL-SHIFT
1131                              <1>     ; 12/04/2021
1132 0000146D 7505                <1>     jnz     short K38A              ; YES, START PROCESSING
1133 0000146F E9AB000000          <1>     jmp     K44                     ; NOT-CTL-SHIFT
1134                              <1>     ;
1135                              <1>     ;----- CONTROL SHIFT, TEST SPECIAL CHARACTERS
1136                              <1>     ;----- TEST FOR BREAK
1137                              <1> K38A:
1138 00001474 3C46                <1>     cmp     al, SCROLL_KEY          ; TEST FOR BREAK
1139 00001476 7530                <1>     jne     short K39               ; JUMP, NO-BREAK
1140 00001478 F6C710              <1>     test    bh, KBX                 ; IS THIS THE ENHANCED KEYBOARD?
1141 0000147B 7405                <1>     jz      short K38B              ; NO, BREAK IS VALID
1142 0000147D F6C702              <1>     test    bh, LC_E0               ; YES, WAS LAST CODE AN E0?
1143 00001480 7426                <1>     jz      short K39               ; NO-BREAK, TEST FOR PAUSE
1144                              <1> K38B:
1145 00001482 8B1D[F6670000]      <1>     mov     ebx, [BUFFER_HEAD]      ; RESET BUFFER TO EMPTY
1146 00001488 891D[FA670000]      <1>     mov     [BUFFER_TAIL], ebx
1147 0000148E C605[E8670000]80    <1>     mov     byte [BIOS_BREAK], 80h  ; TURN ON BIOS_BREAK BIT
1148                              <1>     ;
1149                              <1>     ;----- ENABLE KEYBOARD
1150 00001495 B0AE                <1>     mov     al, ENA_KBD             ; ENABLE KEYBOARD
1151 00001497 E8B5010000          <1>     call    SHIP_IT                 ; EXECUTE ENABLE
1152                              <1>     ;
1153                              <1>     ; CTRL+BREAK code here !!!
1154                              <1>     ;INT    1BH                     ; BREAK INTERRUPT VECTOR
1155                              <1>     ; 17/10/2015
1156 0000149C E88D590000          <1>     call    ctrlbrk ; control+break subroutine
1157                              <1>     ;
1158                              <1>     ;sub    ax, ax                  ; PUT OUT DUMMY CHARACTER
1159                              <1>         ; 12/04/2021
1160 000014A1 29C0                <1>     sub     eax, eax
1161 000014A3 E942010000          <1>         jmp     K57                     ; BUFFER_FILL
1162                              <1>     ;
1163                              <1>     ;----- TEST FOR PAUSE
1164                              <1> K39:                                ; NO_BREAK
1165 000014A8 F6C710              <1>     test    bh, KBX                 ; IS THIS THE ENHANCED KEYBOARD?
1166 000014AB 7537                <1>     jnz     short K41               ; YES, THEN THIS CAN'T BE PAUSE
1167 000014AD 3C45                <1>     cmp     al, NUM_KEY             ; LOOK FOR PAUSE KEY
1168 000014AF 7533                <1>     jne     short K41               ; NO-PAUSE
1169                              <1> K39P:
1170 000014B1 800D[EA670000]08    <1>     or      byte [KB_FLAG_1], HOLD_STATE ; TURN ON THE HOLD FLAG
1171                              <1>     ;
1172                              <1>     ;----- ENABLE KEYBOARD
1173 000014B8 B0AE                <1>     mov     al, ENA_KBD             ; ENABLE KEYBOARD
1174 000014BA E892010000          <1>     call    SHIP_IT                 ; EXECUTE ENABLE
1175                              <1> K39A:
1176 000014BF B020                <1>     mov     al, EOI                 ; END OF INTERRUPT TO CONTROL PORT
1177 000014C1 E620                <1>     out     20h, al ;out INTA00, al       ; ALLOW FURTHER KEYSTROKE INTERRUPTS
1178                              <1>     ;
1179                              <1>     ;----- DURING PAUSE INTERVAL, TURN COLOR CRT BACK ON
1180 000014C3 803D[1E680000]07    <1>         cmp     byte [CRT_MODE], 7      ; IS THIS BLACK AND WHITE CARD
1181 000014CA 740A                <1>         je      short K40               ; YES, NOTHING TO DO
1182 000014CC 66BAD803            <1>     mov     dx, 03D8h               ; PORT FOR COLOR CARD
1183 000014D0 A0[1F680000]        <1>         mov     al, [CRT_MODE_SET]      ; GET THE VALUE OF THE CURRENT MODE
1184 000014D5 EE                  <1>     out     dx, al                  ; SET THE CRT MODE, SO THAT CRT IS ON
1185                              <1>     ;
1186                              <1> K40:                                ; PAUSE-LOOP
1187 000014D6 F605[EA670000]08    <1>         test    byte [KB_FLAG_1], HOLD_STATE ; CHECK HOLD STATE FLAG
1188 000014DD 75F7                <1>     jnz     short K40               ; LOOP UNTIL FLAG TURNED OFF
1189                              <1>     ;
1190 000014DF E995FEFFFF          <1>         jmp     K27                     ; INTERRUPT_RETURN_NO_EOI
1191                              <1>         ;
1192                              <1>     ;----- TEST SPECIAL CASE KEY 55
1193                              <1> K41:                                ; NO-PAUSE
1194 000014E4 3C37                <1>     cmp     al, 55                  ; TEST FOR */PRTSC KEY
1195 000014E6 7513                <1>     jne     short K42               ; NOT-KEY-55
1196 000014E8 F6C710              <1>     test    bh, KBX                 ; IS THIS THE ENHANCED KEYBOARD?
1197 000014EB 7405                <1>     jz      short K41A              ; NO, CTL-PRTSC IS VALID
1198 000014ED F6C702              <1>     test    bh, LC_E0               ; YES, WAS LAST CODE AN E0?
1199 000014F0 7421                <1>     jz      short K42B              ; NO, TRANSLATE TO A FUNCTION
1200                              <1> K41A:
1201 000014F2 66B80072            <1>     mov     ax, 114*256             ; START/STOP PRINTING SWITCH
1202 000014F6 E9EF000000          <1>         jmp     K57                     ; BUFFER_FILL
1203                              <1>     ;
1204                              <1>     ;----- SET UP TO TRANSLATE CONTROL SHIFT
1205                              <1> K42:                                ; NOT-KEY-55
1206 000014FB 3C0F                <1>     cmp     al, 15                  ; IS IT THE TAB KEY?
1207 000014FD 7414                <1>     je      short K42B              ; YES, XLATE TO FUNCTION CODE
1208 000014FF 3C35                <1>     cmp     al, 53                  ; IS IT THE / KEY?
1209 00001501 750E                <1>     jne     short K42A              ; NO, NO MORE SPECIAL CASES
1210 00001503 F6C702              <1>     test    bh, LC_E0               ; YES, IS IT FROM THE KEY PAD?
1211 00001506 7409                <1>     jz      short K42A              ; NO, JUST TRANSLATE
1212 00001508 66B80095            <1>     mov     ax, 9500h               ; YES, SPECIAL CODE FOR THIS ONE
1213 0000150C E9D9000000          <1>     jmp     K57                     ; BUFFER FILL
1214                              <1> K42A:
1215                              <1>     ;;mov   ebx, _K8                ; SET UP TO TRANSLATE CTL
1216 00001511 3C3B                <1>     cmp     al, 59                  ; IS IT IN CHARACTER TABLE?
1217                              <1>         ;jb short K45F                ; YES, GO TRANSLATE CHAR
1218                              <1>     ;;jb    K56 ; 20/02/2015
1219                              <1>     ;;jmp   K64 ; 20/02/2015
1220                              <1> K42B:
1221 00001513 BB[E0660000]        <1>     mov     ebx, _K8                ; SET UP TO TRANSLATE CTL
1222                              <1>     ;jb     K56 ;; 20/02/2015
1223                              <1>     ; 12/04/2021
1224 00001518 7267                <1>     jb      short K45F
1225 0000151A E9B9000000          <1>     jmp     K64
1226                              <1>         ;
1227                              <1>     ;----- NOT IN CONTROL SHIFT
1228                              <1> K44:                                ; NOT-CTL-SHIFT
1229 0000151F 3C37                <1>     cmp     al, 55                  ; PRINT SCREEN KEY?
1230 00001521 7528                <1>     jne     short K45               ; NOT PRINT SCREEN
1231 00001523 F6C710              <1>     test    bh, KBX                 ; IS THIS ENHANCED KEYBOARD?
1232 00001526 7407                <1>     jz      short K44A              ; NO, TEST FOR SHIFT STATE
1233 00001528 F6C702              <1>     test    bh, LC_E0               ; YES, LAST CODE A MARKER?
1234 0000152B 7507                <1>     jnz     short K44B              ; YES, IS PRINT SCREEN
1235 0000152D EB41                <1>     jmp     short K45C              ; NO, TRANSLATE TO '*' CHARACTER
1236                              <1> K44A:
1237 0000152F F6C303              <1>     test    bl, LEFT_SHIFT+RIGHT_SHIFT ; NOT 101 KBD, SHIFT KEY DOWN?
1238 00001532 743C                <1>     jz      short K45C              ; NO, TRANSLATE TO '*' CHARACTER
1239                              <1>     ;
1240                              <1>     ;----- ISSUE INTERRUPT TO INDICATE PRINT SCREEN FUNCTION
1241                              <1> K44B:
1242 00001534 B0AE                <1>     mov     al, ENA_KBD             ; INSURE KEYBOARD IS ENABLED
1243 00001536 E816010000          <1>     call    SHIP_IT                 ; EXECUTE ENABLE
1244 0000153B B020                <1>     mov     al, EOI                 ; END OF CURRENT INTERRUPT
```

```
1245 0000153D E620                <1>     out    20h, al ;out INTA00, al       ; SO FURTHER THINGS CAN HAPPEN
1246                              <1>     ; Print Screen !!!                   ; ISSUE PRINT SCREEN INTERRUPT (INT 05h)
1247                              <1>     ;PUSH  BP                            ; SAVE POINTER
1248                              <1>     ;INT   5H                            ; ISSUE PRINT SCREEN INTERRUPT
1249                              <1>     ;POP   BP                            ; RESTORE POINTER
1250 0000153F 8025[EC670000]FC    <1>        and     byte [KB_FLAG_3], ~(LC_E0+LC_E1) ; ZERO OUT THESE FLAGS
1251 00001546 E92EFEFFFF          <1>        jmp     K27                       ; GO BACK WITHOUT EOI OCCURRING
1252                              <1>     ;
1253                              <1>     ;----- HANDLE IN-CORE KEYS
1254                              <1> K45:                                     ; NOT-PRINT-SCREEN
1255 0000154B 3C3A                <1>     cmp    al, 58                        ; TEST FOR IN-CORE AREA
1256 0000154D 7734                <1>     ja     short K46                     ; JUMP IF NOT
1257 0000154F 3C35                <1>     cmp    al, 53                        ; IS THIS THE '/' KEY?
1258 00001551 7505                <1>     jne    short K45A                    ; NO, JUMP
1259 00001553 F6C702              <1>     test   bh, LC_E0                     ; WAS THE LAST CODE THE MARKER?
1260 00001556 7518                <1>     jnz    short K45C                    ; YES, TRANSLATE TO CHARACTER
1261                              <1> K45A:
1262 00001558 B91A000000          <1>     mov    ecx, 26                       ; LENGHT OF SEARCH
1263 0000155D BF[B6660000]        <1>     mov    edi, K30+10                   ; POINT TO TABLE OF A-Z CHARS
1264 00001562 F2AE                <1>     repne  scasb                         ; IS THIS A LETTER KEY?
1265                              <1>            ; 20/02/2015
1266 00001564 7505                <1>     jne    short K45B                      ; NO, SYMBOL KEY
1267                              <1>     ;
1268 00001566 F6C340              <1>     test   bl, CAPS_STATE                ; ARE WE IN CAPS_LOCK?
1269 00001569 750C                <1>     jnz    short K45D                    ; TEST FOR SURE
1270                              <1> K45B:
1271 0000156B F6C303              <1>     test   bl, LEFT_SHIFT+RIGHT_SHIFT ; ARE WE IN SHIFT STATE?
1272 0000156E 750C                <1>     jnz    short K45E                    ; YES, UPPERCASE
1273                              <1>                                          ; NO, LOWERCASE
1274                              <1> K45C:
1275 00001570 BB[38670000]        <1>     mov    ebx, K10                      ; TRANSLATE TO LOWERCASE LETTERS
1276 00001575 EB51                <1>     jmp    short K56
1277                              <1> K45D:                                        ; ALMOST-CAPS-STATE
1278 00001577 F6C303              <1>     test   bl, LEFT_SHIFT+RIGHT_SHIFT ; CL ON. IS SHIFT ON, TOO?
1279 0000157A 75F4                <1>     jnz    short K45C                    ; SHIFTED TEMP OUT OF CAPS STATE
1280                              <1> K45E:
1281 0000157C BB[90670000]        <1>     mov    ebx, K11                      ; TRANSLATE TO UPPER CASE LETTERS
1282 00001581 EB45                <1> K45F:      jmp    short K56
1283                              <1>     ;
1284                              <1>     ;----- TEST FOR KEYS F1 - F10
1285                              <1> K46:                                     ; NOT IN-CORE AREA
1286 00001583 3C44                <1>     cmp    al, 68                        ; TEST FOR F1 - F10
1287                              <1>     ;ja    short K47                     ; JUMP IF NOT
1288                              <1>     ;jmp   short K53                     ; YES, GO DO FN KEY PROCESS
1289 00001585 7635                <1>     jna    short K53
1290                              <1>     ;
1291                              <1>     ;----- HANDLE THE NUMERIC PAD KEYS
1292                              <1> K47:                                     ; NOT F1 - F10
1293 00001587 3C53                <1>     cmp    al, 83                        ; TEST NUMPAD KEYS
1294 00001589 772D                <1>     ja     short K52                     ; JUMP IF NOT
1295                              <1>     ;
1296                              <1>     ;----- KEYPAD KEYS, MUST TEST NUM LOCK FOR DETERMINATION
1297                              <1> K48:
1298 0000158B 3C4A                <1>     cmp    al, 74                        ; SPECIAL CASE FOR MINUS
1299 0000158D 74ED                <1>     je     short K45E                    ; GO TRANSLATE
1300 0000158F 3C4E                <1>     cmp    al, 78                        ; SPECIAL CASE FOR PLUS
1301 00001591 74E9                <1>     je     short K45E                    ; GO TRANSLATE
1302 00001593 F6C702              <1>     test   bh, LC_E0                     ; IS THIS ONE OFTHE NEW KEYS?
1303 00001596 750A                <1>     jnz    short K49                     ; YES, TRANSLATE TO BASE STATE
1304                              <1>     ;
1305 00001598 F6C320              <1>     test   bl, NUM_STATE                 ; ARE WE IN NUM LOCK
1306 0000159B 7514                <1>     jnz    short K50                     ; TEST FOR SURE
1307 0000159D F6C303              <1>     test   bl, LEFT_SHIFT+RIGHT_SHIFT ; ARE WE IN SHIFT STATE?
1308                              <1>     ;jnz   short K51                     ; IF SHIFTED, REALLY NUM STATE
1309 000015A0 75DA                <1>     jnz    short K45E
1310                              <1>     ;
1311                              <1>     ;----- BASE CASE FOR KEYPAD
1312                              <1> K49:
1313 000015A2 3C4C                <1>     cmp    al, 76                        ; SPECIAL CASE FOR BASE STATE 5
1314 000015A4 7504                <1>     jne    short K49A                    ; CONTINUE IF NOT KEYPAD 5
1315 000015A6 B0F0                <1>     mov    al, 0F0h                      ; SPECIAL ASCII CODE
1316 000015A8 EB40                <1>     jmp    short K57                     ; BUFFER FILL
1317                              <1> K49A:
1318 000015AA BB[38670000]        <1>     mov    ebx, K10                      ; BASE CASE TABLE
1319 000015AF EB27                <1>     jmp    short K64                     ; CONVERT TO PSEUDO SCAN
1320                              <1>     ;
1321                              <1>     ;----- MIGHT BE NUM LOCK, TEST SHIFT STATUS
1322                              <1> K50:                                     ; ALMOST-NUM-STATE
1323 000015B1 F6C303              <1>        test    bl, LEFT_SHIFT+RIGHT_SHIFT
1324 000015B4 75EC                <1>     jnz    short K49                     ; SHIFTED TEMP OUT OF NUM STATE
1325 000015B6 EBC4                <1> K51:jmp    short K45E                    ; REALLY NUM STATE
1326                              <1>     ;
1327                              <1>     ;----- TEST FOR THE NEW KEYS ON WT KEYBOARDS
1328                              <1> K52:                                     ; NOT A NUMPAD KEY
1329 000015B8 3C56                <1>     cmp    al, 86                        ; IS IT THE NEW WT KEY?
1330                              <1>     ;jne   short K53                     ; JUMP IF NOT
1331                              <1>     ;jmp   short K45B                    ; HANDLE WITH REST OF LETTER KEYS
1332 000015BA 74AF                <1>     je     short K45B
1333                              <1>     ;
1334                              <1>     ;----- MUST BE F11 OR F12
1335                              <1> K53:                                     ; F1 - F10 COME HERE, TOO
1336 000015BC F6C303              <1>     test   bl, LEFT_SHIFT+RIGHT_SHIFT ; TEST SHIFT STATE
1337 000015BF 74E1                <1>     jz     short K49                     ; JUMP, LOWER CASE PSEUDO SC'S
1338                              <1>            ; 20/02/2015
1339 000015C1 BB[90670000]        <1>     mov    ebx, K11                      ; UPPER CASE PSEUDO SCAN CODES
1340 000015C6 EB10                <1>     jmp    short K64                     ; TRANSLATE SCAN
1341                              <1>     ;
1342                              <1>     ;----- TRANSLATE THE CHARACTER
1343                              <1> K56:                                     ; TRANSLATE-CHAR
1344 000015C8 FEC8                <1>     dec    al                            ; CONVERT ORIGIN
1345 000015CA D7                  <1>     xlat                                 ; CONVERT THE SCAN CODE TO ASCII
1346 000015CB F605[EC670000]02    <1>     test   byte [KB_FLAG_3], LC_E0       ; IS THIS A NEW KEY?
1347 000015D2 7416                <1>     jz     short K57                     ; NO, GO FILL BUFFER
1348 000015D4 B4E0                <1>     mov    ah, MC_E0                     ; YES, PUT SPECIAL MARKER IN AH
1349 000015D6 EB12                <1>     jmp    short K57                     ; PUT IT INTO THE BUFFER
1350                              <1>     ;
1351                              <1>     ;----- TRANSLATE SCAN FOR PSEUDO SCAN CODES
1352                              <1> K64:                                     ; TRANSLATE-SCAN-ORGD
1353 000015D8 FEC8                <1>     dec    al                            ; CONVERT ORIGIN
1354 000015DA D7                  <1>            xlat                                 ; CTL TABLE SCAN
1355 000015DB 88C4                <1>     mov    ah, al                        ; PUT VALUE INTO AH
1356 000015DD B000                <1>     mov    al, 0                         ; ZERO ASCII CODE
1357 000015DF F605[EC670000]02    <1>     test   byte [KB_FLAG_3], LC_E0       ; IS THIS A NEW KEY?
1358 000015E6 7402                <1>     jz     short K57                     ; NO, GO FILL BUFFER
1359 000015E8 B0E0                <1>     mov    al, MC_E0                     ; YES, PUT SPECIAL MARKER IN AL
1360                              <1>     ;
1361                              <1>     ;----- PUT CHARACTER INTO BUFFER
1362                              <1> K57:                                     ; BUFFER_FILL
1363 000015EA 3CFF                <1>     cmp    al, -1                        ; IS THIS AN IGNORE CHAR
1364 000015EC 7405                <1>     je     short K59                     ; YES, DO NOTHING WITH IT
1365                              <1>     ;je    K26                           ; YES, DO NOTHING WITH IT
1366 000015EE 80FCFF              <1>     cmp    ah, -1                        ; LOOK FOR -1 PSEUDO SCAN
1367                              <1>        ;;jne      short K61                    ; NEAR_INTERRUPT_RETURN
1368                              <1>     ;je    K26                           ; INTERRUPT_RETURN
```

```
1369                                    <1>     ; 12/04/2021
1370 000015F1 7505                      <1>         jne short _K60              ; NEAR_INTERRUPT_RETURN
1371                                    <1> K59:                                ; NEAR_INTERRUPT_RETURN
1372 000015F3 E975FDFFFF                <1>     jmp    K26                      ; INTERRUPT_RETURN
1373                                    <1>
1374                                    <1> _K60: ; 29/01/2016
1375 000015F8 80FC68                    <1>     cmp    ah, 68h ; ALT + F1 key
1376 000015FB 721D                      <1>     jb     short K61
1377 000015FD 80FC6F                    <1>     cmp    ah, 6Fh ; ALT + F8 key
1378 00001600 7718                      <1>     ja     short K61
1379                                    <1>     ;
1380 00001602 8A1D[AE760100]            <1>     mov    bl, [ACTIVE_PAGE]
1381 00001608 80C368                    <1>     add    bl, 68h
1382 0000160B 38E3                      <1>     cmp    bl, ah
1383 0000160D 740B                      <1>     je     short K61
1384                                    <1>     ; 24/07/2022
1385                                    <1>     ;push   ax
1386 0000160F 50                        <1>     push   eax
1387 00001610 88E0                      <1>     mov    al, ah
1388 00001612 2C68                      <1>     sub    al, 68h
1389 00001614 E896090000                <1>     call   set_active_page
1390 00001619 58                        <1>     pop    eax
1391                                    <1>     ;pop    ax
1392                                    <1> K61:                                ; NOT-CAPS-STATE
1393 0000161A 8B1D[FA670000]            <1>     mov    ebx, [BUFFER_TAIL]       ; GET THE END POINTER TO THE BUFFER
1394 00001620 89DE                      <1>     mov    esi, ebx                 ; SAVE THE VALUE
1395 00001622 E8B5FAFFFF                <1>     call   _K4                      ; ADVANCE THE TAIL
1396 00001627 3B1D[F6670000]            <1>     cmp    ebx, [BUFFER_HEAD]       ; HAS THE BUFFER WRAPPED AROUND
1397 0000162D 740E                      <1>     je     short K62                ; BUFFER_FULL_BEEP
1398 0000162F 668906                    <1>     mov    [esi], ax                ; STORE THE VALUE
1399 00001632 891D[FA670000]            <1>     mov    [BUFFER_TAIL], ebx       ; MOVE THE POINTER UP
1400 00001638 E930FDFFFF                <1>     jmp    K26
1401                                    <1>     ;;cli                           ; TURN OFF INTERRUPTS
1402                                    <1>     ;;mov  al, EOI                  ; END OF INTERRUPT COMMAND
1403                                    <1>     ;;out  INTA00, al               ; SEND COMMAND TO INTERRUPT CONTROL PORT
1404                                    <1>     ;mov   al, ENA_KBD              ; INSURE KEYBOARD IS ENABLED
1405                                    <1>     ;call  SHIP_IT                  ; EXECUTE ENABLE
1406                                    <1>     ;mov   ax, 9102h                ; MOVE IN POST CODE & TYPE
1407                                    <1>     ;int   15h                      ; PERFORM OTHER FUNCTION
1408                                    <1>     ;;and  byte [KB_FLAG_3],~(LC_E0+LC_E1) ; RESET LAST CHAR H.C. FLAG
1409                                    <1>     ;jmp   K27A                     ; INTERRUPT_RETURN
1410                                    <1>     ;;jmp   K27
1411                                    <1>     ;
1412                                    <1>     ;-----  BUFFER IS FULL SOUND THE BEEPER
1413                                    <1> K62:
1414 0000163D B020                      <1>     mov    al, EOI                  ; ENABLE INTERRUPT CONTROLLER CHIP
1415 0000163F E620                      <1>     out    INTA00, al
1416 00001641 66B9A602                  <1>     mov    cx, 678                  ; DIVISOR FOR 1760 HZ
1417 00001645 B304                      <1>     mov    bl, 4                    ; SHORT BEEP COUNT (1/16 + 1/64 DELAY)
1418 00001647 E8880D0000                <1>     call   beep                     ; GO TO COMMON BEEP HANDLER
1419 0000164C E928FDFFFF                <1>     jmp     K27                     ; EXIT
1420                                    <1>
1421                                    <1> SHIP_IT:
1422                                    <1>     ;--------------------------------------------------------------------------------
1423                                    <1>     ; SHIP_IT
1424                                    <1>     ;      THIS ROUTINES HANDLES TRANSMISSION OF COMMAND AND DATA BYTES
1425                                    <1>     ;      TO THE KEYBOARD CONTROLLER.
1426                                    <1>     ;--------------------------------------------------------------------------------
1427                                    <1>     ;
1428                                    <1>
1429                                    <1>     ;push   ax                      ; SAVE DATA TO SEND
1430                                    <1>     ; 12/04/2021
1431 00001651 50                        <1>     push   eax
1432                                    <1>
1433                                    <1>     ;-----  WAIT FOR COMMAND TO ACCEPTED
1434 00001652 FA                        <1>     cli                             ; DISABLE INTERRUPTS TILL DATA SENT
1435                                    <1>     ; xor   ecx, ecx                ; CLEAR TIMEOUT COUNTER
1436 00001653 B900000100                <1>     mov    ecx, 10000h
1437                                    <1> S10:
1438 00001658 E464                      <1>     in     al, STATUS_PORT                ; READ KEYBOARD CONTROLLER STATUS
1439 0000165A A802                      <1>     test   al, INPT_BUF_FULL        ; CHECK FOR ITS INPUT BUFFER BUSY
1440 0000165C E0FA                      <1>     loopnz S10                      ; WAIT FOR COMMAND TO BE ACCEPTED
1441                                    <1>
1442                                    <1>     ;pop    ax                      ; GET DATA TO SEND
1443                                    <1>     ; 12/04/2021
1444 0000165E 58                        <1>     pop    eax
1445                                    <1>
1446 0000165F E664                      <1>     out    STATUS_PORT, al                ; SEND TO KEYBOARD CONTROLLER
1447 00001661 FB                        <1>     sti                             ; ENABLE INTERRUPTS AGAIN
1448 00001662 C3                        <1>     retn                            ; RETURN TO CALLER
1449                                    <1>
1450                                    <1>     ; 12/04/2021 (32 bit push/pop)
1451                                    <1> SND_DATA:
1452                                    <1>     ; ------------------------------------------------------------------------------
1453                                    <1>     ; SND_DATA
1454                                    <1>     ;      THIS ROUTINES HANDLES TRANSMISSION OF COMMAND AND DATA BYTES
1455                                    <1>     ;      TO THE KEYBOARD AND RECEIPT OF ACKNOWLEDGEMENTS. IT ALSO
1456                                    <1>     ;      HANDLES ANY RETRIES IF REQUIRED
1457                                    <1>     ; ------------------------------------------------------------------------------
1458                                    <1>     ;
1459 00001663 50                        <1>     push   eax ; push ax            ; SAVE REGISTERS
1460 00001664 53                        <1>     push   ebx ; push bx
1461 00001665 51                        <1>     push   ecx
1462 00001666 88C7                      <1>     mov    bh, al                   ; SAVE TRANSMITTED BYTE FOR RETRIES
1463 00001668 B303                      <1>     mov    bl, 3                    ; LOAD RETRY COUNT
1464                                    <1> SD0:
1465 0000166A FA                        <1>     cli                             ; DISABLE INTERRUPTS
1466 0000166B 8025[EB670000]CF          <1>     and    byte [KB_FLAG_2], ~(KB_FE+KB_FA) ; CLEAR ACK AND RESEND FLAGS
1467                                    <1>     ;
1468                                    <1>     ;-----  WAIT FOR COMMAND TO BE ACCEPTED
1469 00001672 B900000100                <1>     mov    ecx, 10000h              ; MAXIMUM WAIT COUNT
1470                                    <1> SD5:
1471 00001677 E464                      <1>     in     al, STATUS_PORT                ; READ KEYBOARD PROCESSOR STATUS PORT
1472 00001679 A802                      <1>     test   al, INPT_BUF_FULL        ; CHECK FOR ANY PENDING COMMAND
1473 0000167B E0FA                      <1>     loopnz SD5                      ; WAIT FOR COMMAND TO BE ACCEPTED
1474                                    <1>     ;
1475 0000167D 88F8                      <1>     mov    al, bh                   ; REESTABLISH BYTE TO TRANSMIT
1476 0000167F E660                      <1>     out    PORT_A, al               ; SEND BYTE
1477 00001681 FB                        <1>     sti                             ; ENABLE INTERRUPTS
1478                                    <1>     ;mov   cx, 01A00h               ; LOAD COUNT FOR 10 ms+
1479 00001682 B9FFFF0000                <1>     mov    ecx, 0FFFFh
1480                                    <1> SD1:
1481 00001687 F605[EB670000]30          <1>     test   byte [KB_FLAG_2], KB_FE+KB_FA ; SEE IF EITHER BIT SET
1482 0000168E 750F                      <1>     jnz    short SD3                ; IF SET, SOMETHING RECEIVED GO PROCESS
1483 00001690 E2F5                      <1>     loop   SD1                      ; OTHERWISE WAIT
1484                                    <1> SD2:
1485 00001692 FECB                      <1>     dec    bl                       ; DECREMENT RETRY COUNT
1486 00001694 75D4                      <1>     jnz    short SD0                ; RETRY TRANSMISSION
1487 00001696 800D[EB670000]80          <1>     or     byte [KB_FLAG_2], KB_ERR ; TURN ON TRANSMIT ERROR FLAG
1488 0000169D EB09                      <1>     jmp    short SD4                ; RETRIES EXHAUSTED FORGET TRANSMISSION
1489                                    <1> SD3:
1490 0000169F F605[EB670000]10          <1>     test   byte [KB_FLAG_2], KB_FA ; SEE IF THIS IS AN ACKNOWLEDGE
1491 000016A6 74EA                      <1>     jz     short SD2                ; IF NOT, GO RESEND
1492                                    <1> SD4:
```

```
1493 000016A8 59                <1>     pop     ecx                     ; RESTORE REGISTERS
1494 000016A9 5B                <1>     pop     ebx ; pop bx
1495 000016AA 58                <1>     pop     eax ; pop ax
1496 000016AB C3                <1>     retn                            ; RETURN, GOOD TRANSMISSION
1497                            <1>
1498                            <1> SND_LED:
1499                            <1>     ; ------------------------------------------------------------------------------
1500                            <1>     ; SND_LED
1501                            <1>     ;       THIS ROUTINES TURNS ON THE MODE INDICATORS.
1502                            <1>     ;
1503                            <1>     ;------------------------------------------------------------------------------
1504                            <1>     ;
1505 000016AC FA                <1>     cli                             ; TURN OFF INTERRUPTS
1506 000016AD F605[EB670000]40  <1>     test    byte [KB_FLAG_2], KB_PR_LED ; CHECK FOR MODE INDICATOR UPDATE
1507 000016B4 755F              <1>     jnz     short SL1               ; DON'T UPDATE AGAIN IF UPDATE UNDERWAY
1508                            <1>     ;
1509 000016B6 800D[EB670000]40  <1>     or      byte [KB_FLAG_2], KB_PR_LED ; TURN ON UPDATE IN PROCESS
1510 000016BD B020              <1>     mov     al, EOI                 ; END OF INTERRUPT COMMAND
1511 000016BF E620              <1>     out     20h, al ;out INTA00, al       ; SEND COMMAND TO INTERRUPT CONTROL PORT
1512 000016C1 EB11              <1>     jmp     short SL0               ; GO SEND MODE INDICATOR COMMAND
1513                            <1> SND_LED1:
1514 000016C3 FA                <1>     cli                             ; TURN OFF INTERRUPTS
1515 000016C4 F605[EB670000]40  <1>     test    byte [KB_FLAG_2], KB_PR_LED ; CHECK FOR MODE INDICATOR UPDATE
1516 000016CB 7548              <1>     jnz     short SL1               ; DON'T UPDATE AGAIN IF UPDATE UNDERWAY
1517                            <1>     ;
1518 000016CD 800D[EB670000]40  <1>     or      byte [KB_FLAG_2], KB_PR_LED ; TURN ON UPDATE IN PROCESS
1519                            <1> SL0:
1520 000016D4 B0ED              <1>     mov     al, LED_CMD             ; LED CMD BYTE
1521 000016D6 E888FFFFFF        <1>     call    SND_DATA                ; SEND DATA TO KEYBOARD
1522 000016DB FA                <1>     cli
1523 000016DC E836000000        <1>     call    MAKE_LED                ; GO FORM INDICATOR DATA BYTE
1524 000016E1 8025[EB670000]F8  <1>     and     byte [KB_FLAG_2], 0F8h; ~KB_LEDS ; CLEAR MODE INDICATOR BITS
1525 000016E8 0805[EB670000]    <1>     or      [KB_FLAG_2], al         ; SAVE PRESENT INDICATORS FOR NEXT TIME
1526 000016EE F605[EB670000]80  <1>     test    byte [KB_FLAG_2], KB_ERR ; TRANSMIT ERROR DETECTED
1527 000016F5 750F              <1>     jnz     short SL2               ; IF SO, BYPASS SECOND BYTE TRANSMISSION
1528                            <1>     ;
1529 000016F7 E867FFFFFF        <1>     call    SND_DATA                ; SEND DATA TO KEYBOARD
1530 000016FC FA                <1>     cli                             ; TURN OFF INTERRUPTS
1531 000016FD F605[EB670000]80  <1>     test    byte [KB_FLAG_2], KB_ERR ; TRANSMIT ERROR DETECTED
1532 00001704 7408              <1>     jz      short SL3               ; IF NOT, DON'T SEND AN ENABLE COMMAND
1533                            <1> SL2:
1534 00001706 B0F4              <1>     mov     al, KB_ENABLE           ; GET KEYBOARD CSA ENABLE COMMAND
1535 00001708 E856FFFFFF        <1>     call    SND_DATA                ; SEND DATA TO KEYBOARD
1536 0000170D FA                <1>     cli                             ; TURN OFF INTERRUPTS
1537                            <1> SL3:
1538 0000170E 8025[EB670000]3F  <1>     and     byte [KB_FLAG_2], ~(KB_PR_LED+KB_ERR) ; TURN OFF MODE INDICATOR
1539                            <1> SL1:                                ; UPDATE AND TRANSMIT ERROR FLAG
1540 00001715 FB                <1>     sti                             ; ENABLE INTERRUPTS
1541 00001716 C3                <1>     retn                            ; RETURN TO CALLER
1542                            <1>
1543                            <1> MAKE_LED:
1544                            <1>     ;------------------------------------------------------------------------------
1545                            <1>     ; MAKE_LED
1546                            <1>     ;       THIS ROUTINES FORMS THE DATA BYTE NECESSARY TO TURN ON/OFF
1547                            <1>     ;       THE MODE INDICATORS.
1548                            <1>     ;------------------------------------------------------------------------------
1549                            <1>     ;
1550                            <1>     ;push   cx                      ; SAVE CX
1551 00001717 A0[E9670000]      <1>     mov     al, [KB_FLAG]           ; GET CAPS & NUM LOCK INDICATORS
1552 0000171C 2470              <1>     and     al, CAPS_STATE+NUM_STATE+SCROLL_STATE ; ISOLATE INDICATORS
1553                            <1>     ;mov    cl, 4                   ; SHIFT COUNT
1554                            <1>     ;rol    al, cl                  ; SHIFT BITS OVER TO TURN ON INDICATORS
1555 0000171E C0C004            <1>     rol     al, 4 ; 20/02/2015
1556 00001721 2407              <1>     and     al, 07h                 ; MAKE SURE ONLY MODE BITS ON
1557                            <1>     ;pop    cx
1558 00001723 C3                <1>     retn                            ; RETURN TO CALLER
1559                            <1>
1560                            <1> ; % include 'kybdata.s'   ; KEYBOARD DATA
1561                            <1>
1562                            <1>
1563                            <1> ; /// End Of KEYBOARD FUNCTIONS ///
2989
2990                                %include 'video.s' ; 07/03/2015
   1                            <1> ; ****************************************************************************
   2                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.10 - video.s
   3                            <1> ; ----------------------------------------------------------------------------
   4                            <1> ; Last Update: 22/12/2025 (Previous: 29/11/2023 - Kernel v2.0.7)
   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 00001724 9C                <1>     pushfd
  35 00001725 0E                <1>     push    cs
  36 00001726 E851000000        <1>     call    VIDEO_IO_1
  37 0000172B 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                           :
```

```
                        <1> ;           (AL) = 06H  640X200 BW MODE                                     :
                        <1> ;           (AL) = 07H   80X25 MONOCHROME (USED INTERNAL TO VIDEO ONLY)       :
                        <1> ;           *** NOTES -BW MODES OPERATE SAME AS COLOR MODES, BUT COLOR :
                        <1> ;                      BURST IS NOT ENABLED                                 :
                        <1> ;                     -CURSOR IS NOT DISPLAYED IN GRAPHICS MODE        :
                        <1> ;    (AH)= 01H     SET CURSOR TYPE                                                 :
                        <1> ;           (CH) = BITS 4-0 = START LINE FOR CURSOR                           :
                        <1> ;                  ** HARDWARE WILL ALWAYS CAUSE BLINK                 :
                        <1> ;                  ** SETTING BIT 5 OR 6 WILL CAUSE ERRATIC BLINKING   :
                        <1> ;                     OR NO CURSOR AT ALL                              :
                        <1> ;           (CL) = BITS 4-0 = END LINE FOR CURSOR                             :
                        <1> ;    (AH)= 02H     SET CURSOR POSITION                                       :
                        <1> ;           (DH,DL) = ROW,COLUMN  (00H,00H) IS UPPER LEFT                     :
                        <1> ;           (BH) = A PAGE NUMBER (MUST BE 00H FOR GRAPHICS MODES)             :
                        <1> ;    (AH)= 03H     READ CURSOR POSITION                                      :
                        <1> ;           (BH) = PAGE NUMBER (MUST BE 00H FOR GRAPHICS MODES)        :
                        <1> ;           ON EXIT (DH,DL) = ROW,COLUMN OF CURRENT CURSOR                    :
                        <1> ;                   (CH,CL) = CURSOR MODE CURRENTLY SET                  :
                        <1> ;    (AH)= 04H     READ LIGHT PEN POSITION                                         :
                        <1> ;           ON EXIT:                                                     :
                        <1> ;           (AH) = 00H -- LIGHT PEN SWITCH NOT DOWN/NOT TRIGGERED             :
                        <1> ;           (AH) = 01H -- VALID LIGHT PEN VALUE IN REGISTERS             :
                        <1> ;                   (DH,DL) = ROW,COLUMN OF CHARACTER LP POSITION        :
                        <1> ;                   (CH) = RASTER LINE (0-199)                           :
                        <1> ;                   (BX) = PIXEL COLUMN (0-319,639)                      :
                        <1> ;    (AH)= 05H     SELECT ACTIVE DISPLAY PAGE (VALID ONLY FOR ALPHA MODES)          :
                        <1> ;           (AL) = NEW PAGE VALUE (0-7 FOR MODES 0&1, 0-3 FOR MODES 2&3)      :
                        <1> ;    (AH)= 06H     SCROLL ACTIVE PAGE UP                                     :
                        <1> ;           (AL) = NUMBER OF LINES. ( LINES BLANKED AT BOTTOM OF WINDOW )      :
                        <1> ;                   (AL) = 00H MEANS BLANK ENTIRE WINDOW                 :
                        <1> ;           (CH,CL) = ROW,COLUMN OF UPPER LEFT CORNER OF SCROLL               :
                        <1> ;           (DH,DL) = ROW,COLUMN OF LOWER RIGHT CORNER OF SCROLL              :
                        <1> ;           (BH) = ATTRIBUTE TO BE USED ON BLANK LINE                         :
                        <1> ;    (AH)= 07H     SCROLL ACTIVE PAGE DOWN                                         :
                        <1> ;           (AL) = NUMBER OF LINES, INPUT LINES BLANKED AT TOP OF WINDOW       :
                        <1> ;                   (AL) = 00H MEANS BLANK ENTIRE WINDOW                 :
                        <1> ;           (CH,CL) = ROW,COLUMN OF UPPER LEFT CORNER OF SCROLL               :
                        <1> ;           (DH,DL) = ROW,COLUMN OF LOWER RIGHT CORNER OF SCROLL                :
                        <1> ;           (BH) = ATTRIBUTE TO BE USED ON BLANK LINE                         :
                        <1> ;                                                                             :
                        <1> ;    CHARACTER HANDLING ROUTINES                                                :
                        <1> ;                                                                             :
                        <1> ;     (AH)= 08H     READ ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION         :
                        <1> ;           (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)                  :
                        <1> ;           ON EXIT:                                                          :
                        <1> ;           (AL) = CHAR READ                                                  :
                        <1> ;           (AH) = ATTRIBUTE OF CHARACTER READ (ALPHA MODES ONLY)              :
                        <1> ;     (AH)= 09H     WRITE ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION               :
                        <1> ;           (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)                  :
                        <1> ;           (CX) = COUNT OF CHARACTERS TO WRITE                               :
                        <1> ;           (AL) = CHAR TO WRITE                                              :
                        <1> ;           (BL) = ATTRIBUTE OF CHARACTER (ALPHA)/COLOR OF CHAR (GRAPHICS)    :
                        <1> ;                    SEE NOTE ON WRITE DOT FOR BIT 7 OF BL = 1.               :
                        <1> ;     (AH) = 0AH    WRITE CHARACTER ONLY AT CURRENT CURSOR POSITION                 :
                        <1> ;           (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)                  :
                        <1> ;           (CX) = COUNT OF CHARACTERS TO WRITE                               :
                        <1> ;           (AL) = CHAR TO WRITE                                              :
                        <1> ;                  NOTE: USE FUNCTION (AH)= 09H IN GRAPHICS MODES             :
                        <1> ;    FOR READ/WRITE CHARACTER INTERFACE WHILE IN GRAPHICS MODE, THE             :
                        <1> ;           CHARACTERS ARE FORMED FROM A CHARACTER GENERATOR IMAGE             :
                        <1> ;           MAINTAINED IN THE SYSTEM ROM. ONLY THE 1ST 128 CHARS               :
                        <1> ;           ARE CONTAINED THERE. TO READ/WRITE THE SECOND 128 CHARS,   :
                        <1> ;           THE USER MUST INITIALIZE THE POINTER AT INTERRUPT 1FH              :
                        <1> ;           (LOCATION 0007CH) TO POINT TO THE 1K BYTE TABLE CONTAINING :
                        <1> ;           THE CODE POINTS FOR THE SECOND 128 CHARS (128-255).        :
                        <1> ;    FOR WRITE CHARACTER INTERFACE IN GRAPHICS MODE, THE REPLICATION FACTOR    :
                        <1> ;           CONTAINED IN (CX) ON ENTRY WILL PRODUCE VALID RESULTS ONLY :
                        <1> ;           FOR CHARACTERS CONTAINED ON THE SAME ROW. CONTINUATION TO  :
                        <1> ;           SUCCEEDING LINES WILL NOT PRODUCE CORRECTLY.                :
                        <1> ;                                                                      :
                        <1> ;     GRAPHICS INTERFACE                                                      :
                        <1> ;     (AH)= 0BH     SET COLOR PALETTE                                         :
                        <1> ;           (BH) = PALETTE COLOR ID BEING SET (0-127)                  :
                        <1> ;           (BL) = COLOR VALUE TO BE USED WITH THAT COLOR ID           :
                        <1> ;                  NOTE: FOR THE CURRENT COLOR CARD, THIS ENTRY POINT HAS     :
                        <1> ;                          MEANING ONLY FOR 320X200 GRAPHICS.         :
                        <1> ;                  COLOR ID = 0 SELECTS THE BACKGROUND COLOR (0-15)            :
                        <1> ;                  COLOR ID = 1 SELECTS THE PALETTE TO BE USED:        :
                        <1> ;                          0 = GREEN(1)/RED(2)/YELLOW(3)              :
                        <1> ;                          1 = CYAN(1)/MAGENTA(2)/WHITE(3)                 :
                        <1> ;                  IN 40X25 OR 80X25 ALPHA MODES, THE VALUE SET FOR    :
                        <1> ;                          PALETTE COLOR 0 INDICATES THE BORDER COLOR  :
                        <1> ;                          TO BE USED (VALUES 0-31, WHERE 16-31 SELECT :
                        <1> ;                          THE HIGH INTENSITY BACKGROUND SET.          :
                        <1> ;     (AH)= 0CH     WRITE DOT                                                 :
                        <1> ;           (DX) = ROW NUMBER                                          :
                        <1> ;           (CX) = COLUMN NUMBER                                       :
                        <1> ;           (AL) = COLOR VALUE                                         :
                        <1> ;                   IF BIT 7 OF AL = 1, THEN THE COLOR VALUE IS EXCLUSIVE     :
                        <1> ;                   ORed WITH THE CURRENT CONTENTS OF THE DOT          :
                        <1> ;     (AH)= 0DH     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>
0000172C [431B0000]     <1> M1: dd      SET_MODE        ; TABLE OF ROUTINES WITHIN VIDEO I/O
00001730 [F91E0000]     <1>     dd      SET_CTYPE
00001734 [271F0000]     <1>     dd      SET_CPOS
00001738 [4B1F0000]     <1>     dd      READ_CURSOR
                        <1>     ;dd     VIDEO_RETURN    ; READ_LPEN
0000173C [411B0000]     <1>     dd      set_mode_ncm    ; Set mode without clearing video memory
00001740 [911F0000]     <1>     dd      ACT_DISP_PAGE
00001744 [1B200000]     <1>     dd      SCROLL_UP
00001748 [32210000]     <1>     dd      SCROLL_DOWN
0000174C [B1210000]     <1>     dd      READ_AC_CURRENT
00001750 [0E220000]     <1>     dd      WRITE_AC_CURRENT
00001754 [34220000]     <1>     dd      WRITE_C_CURRENT
00001758 [3C2B0000]     <1>     dd      SET_COLOR
0000175C [A42B0000]     <1>     dd      WRITE_DOT
00001760 [732B0000]     <1>     dd      READ_DOT
00001764 [BE220000]     <1>     dd      WRITE_TTY
00001768 [291B0000]     <1>     dd      VIDEO_STATE
0000176C [71360000]     <1>     dd      vga_pal_funcs  ; 10/08/2016 (TRDOS 386)
00001770 [89300000]     <1>     dd      font_setup     ; 10/07/2016 (TRDOS 386)
00001774 [781B0000]     <1>     dd      VIDEO_RETURN   ; RESERVED
00001778 [34240000]     <1>     dd      WRITE_STRING   ; 23/06/2016 (TRDOS 386)
                        <1> M1L EQU     $ - M1
                        <1>
                        <1> ; 22/12/2025
                        <1> ; 21/12/2025 - TRDOS 386 v2.0.10
                        <1> ; 29/11/2023 - TRDOS 386 v2.0.7
                        <1> ;   [VESA VBE3-PMI functions]
                        <1> ;   (fixing page table problems for LFB/PMI for >2.5GB main memory)
                        <1> ;   ((Addresses of kernel page tables and PMI videobios address
                        <1> ;     must be < 4MB or cr3 register must contain kernel's page dir
                        <1> ;     during video interrupt. Save-change-restore cr3 register
                        <1> ;     content here is a bugfix method to prevent page faults
                        <1> ;     for -real- computers with >2.5GB main memory.))
                        <1> ;    {INT 31h was not changing cr3 while it contains user's
                        <1> ;     page directory addr and accessing beyond of the 1st 4MB was
                        <1> ;     causing to page faults.. because, after the 1st 4MB, user's
                        <1> ;     page tables contain non-linear/virtual memory pages.}
                        <1> ;
                        <1> ; 02/08/2022 - TRDOS 386 v2.0.5
                        <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
0000177C 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
0000177D 80FC14         <1>     cmp     ah, M1L/4
00001780 7205           <1>     jb      short VGA_func
                        <1>
00001782 80FC4F         <1>     cmp     ah, 4Fh
00001785 7536           <1>     jne     short M4 ; invalid !
                        <1>
                        <1> VGA_func: ; 26/11/2020
00001787 06             <1>     push    es ; *
00001788 1E             <1>     push    ds ; **                 ; SAVE WORK AND PARAMETER REGISTERS
                        <1>
                        <1>     ; 26/11/2020
00001789 50             <1>     push    eax ; -
                        <1>
0000178A 66B81000       <1>     mov     ax, KDATA               ; POINT DS: TO DATA SEGMENT
0000178E 8ED8           <1>     mov     ds, ax
00001790 8EC0           <1>     mov     es, ax
                        <1>
                        <1>     ; 26/11/2020
00001792 58             <1>     pop     eax ; +
                        <1>     ;
00001793 FB             <1>     sti
00001794 80FC4F         <1>     cmp     ah, 4Fh
                        <1>     ;je     short VBE_func
                        <1>     ; 21/12/2025
00001797 0F84BC000000   <1>     je      VBE_func
                        <1>
                        <1>     ; 04/12/2020
0000179D A3[08830100]   <1>     mov     [video_eax], eax
                        <1>
                        <1>     ; 21/12/2020
000017A2 803D[1E680000]FF <1>   cmp     byte [CRT_MODE], 0FFh ; Current mode is a VESA VBE mode ?
000017A9 7213           <1>     jb      short VGA_func_std
                        <1>
000017AB 08E4           <1>     or      ah, ah                  ; set mode ?
000017AD 750F           <1>     jnz     short VGA_func_std      ; no
                        <1>
000017AF 803D[86090000]03 <1>   cmp     byte [vbe3], 3          ; (real) VESA VBE 3 video bios ?
000017B6 7506           <1>     jne     short VGA_func_std      ; no
                        <1>
000017B8 E9D7010000     <1>     jmp     vesa_vbe3_pmi
                        <1>
                        <1>     ; 21/12/2020
                        <1> M4:                                 ; COMMAND NOT VALID
000017BD CF             <1>     iretd                           ; DO NOTHING IF NOT IN VALID RANGE
                        <1>
                        <1> VGA_func_std:
                        <1>     ; 06/12/2020
```

```
                                <1>     ; 03/12/2020
000017BE 80FC0F                 <1>     cmp     ah, 0Fh
                                <1>     ;ja     short VGA_funcs_0     ; only CGA funcs will be handled by
                                <1>     ; 21/12/2025
000017C1 777F                   <1>     ja      VGA_funcs_0           ; vga bios firmware
                                <1>
                                <1>     ; 06/12/2020
                                <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
000017C3 803D[EC9C0100]00       <1>     cmp     byte [pmi32], 0             ; 32 bit protected mode interface for
000017CA 7676                   <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
000017CC 20E4                   <1>     and     ah, ah ; is this set mode ?
000017CE 7413                   <1>     jz      short vbe_pmi32_2 ; yes
                                <1>
000017D0 80FC04                 <1>     cmp     ah, 04h  ; set mode ('no clear memory' option)
000017D3 746D                   <1>     je      short VGA_funcs_0
                                <1>
                                <1>     ; 07/12/2020
000017D5 803D[1E680000]07       <1>     cmp     byte [CRT_MODE], 7 ; current mode > 7 ?
000017DC 7664                   <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:
000017DE E9B1010000             <1>     jmp     vesa_vbe3_pmi
                                <1>
                                <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>
                                <1>     ; 22/12/2025
                                <1>     ; 21/12/2025 - TRDOS 386 v2.0.10
                                <1>     ; set character height (needed) for next system calls
                                <1>     ;;;;
                                <1> vbe_pmi32_3:
000017E3 56                     <1>     push    esi
000017E4 50                     <1>     push    eax
000017E5 88C4                   <1>     mov     ah, al
000017E7 80E47F                 <1>     and     ah, 7Fh
000017EA BE[3A680000]           <1>     mov     esi, vga_modes
                                <1> vbe_pmi32_3_1:
000017EF AC                     <1>     lodsb
000017F0 38C4                   <1>     cmp     ah, al
000017F2 7408                   <1>     je      short vbe_pmi32_3_2
000017F4 81FE[4A680000]         <1>     cmp     esi, vga_modes+vga_mode_count
000017FA 72F3                   <1>     jb      short vbe_pmi32_3_1
                                <1> vbe_pmi32_3_2:
000017FC 81EE[3A680000]         <1>     sub     esi, vga_modes
00001802 C1E602                 <1>     shl     esi, 2  ; dword
00001805 81C6[4A680000]         <1>     add     esi, vga_mode_tbl_ptr
0000180B 8B36                   <1>     mov     esi, [esi]
0000180D 66AD                   <1>     lodsw
0000180F A2[20680000]           <1>     mov     [CRT_COLS], al
00001814 FEC4                   <1>     inc     ah
00001816 8825[26680000]         <1>     mov     [VGA_ROWS], ah
0000181C 8A06                   <1>     mov     al, [esi]
                                <1>     ;lodsb
0000181E A2[22680000]           <1>     mov     [CHAR_HEIGHT], al
                                <1> vbe_pmi32_4:
00001823 58                     <1>     pop     eax
00001824 5E                     <1>     pop     esi
                                <1>     ;;;;
                                <1>
                                <1>     ; 07/12/2020
00001825 803D[1E680000]07       <1>     cmp     byte [CRT_MODE], 7 ; current mode > 7 ?
0000182C 77B0                   <1>     ja      short vbe_pmi32_1  ; yes
                                <1>
0000182E 3C07                   <1>     cmp     al, 7   ; requested mode > 7 ?
00001830 7610                   <1>     jna     short VGA_funcs_0  ; no (CGA)
                                <1>
00001832 3C13                   <1>     cmp     al, 13h
00001834 76A8                   <1>     jna     short vbe_pmi32_1
                                <1>
00001836 A880                   <1>     test    al, 80h
00001838 7408                   <1>     jz      short VGA_funcs_0  ; unknown or special
                                <1>
0000183A 3C87                   <1>     cmp     al, 87h ; requested mode > 7 ?
                                <1>                     ; (with no clear mem ops)
0000183C 7604                   <1>     jna     short VGA_funcs_0  ; no (CGA)
                                <1>
0000183E 3C93                   <1>     cmp     al, 93h
00001840 769C                   <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:
00001842 52                     <1>     push    edx ; ***
00001843 51                     <1>     push    ecx ; ****
00001844 53                     <1>     push    ebx ; *****
00001845 56                     <1>     push    esi ; ******
00001846 57                     <1>     push    edi ; *******
00001847 55                     <1>     push    ebp ; ********
                                <1>
                                <1>     ;mov    [video_eax], eax ; 12/05/2016
00001848 BF00800B00             <1>     mov     edi, 0B8000h          ; GET offset FOR COLOR CARD
                                <1>
                                <1>     ; 23/03/2016
```

```
424 0000184D C0E402             <1>     shl     ah, 2  ; dword          ; TIMES 2 FOR WORD TABLE LOOKUP
425 00001850 0FB6F4             <1>     movzx   esi, ah                 ; MOVE OFFSET INTO LOOK UP REGISTER (SI)
426                             <1>     ;mov    ah, [CRT_MODE]          ; MOVE CURRENT MODE INTO (AH) REGISTER
427                             <1>
428                             <1>     ;;15/01/2017
429                             <1>     ; 14/01/2017
430                             <1>     ; 02/01/2017
431                             <1>     ;;mov   byte [intflg], 31h  ; video interrupt
432                             <1>     ;sti ; 26/11/2020
433                             <1>     ;
434                             <1>
435 00001853 FFA6[2C170000]     <1>     jmp     dword [esi+M1]          ; GO TO SELECTED FUNCTION
436                             <1>
437                             <1> VBE_func:
438                             <1>     ; 26/11/2020
439                             <1>     ;sti
440 00001859 55                 <1>     push    ebp ; *** ; 27/11/2020
441 0000185A 56                 <1>     push    esi ; ****
442                             <1>
443                             <1>     ; Note:
444                             <1>     ; ebx, ecx, edx, edi, ebp registers
445                             <1>     ; must be saved by VBE functions and
446                             <1>     ; eax register must be set
447                             <1>     ; (according to VBE 3 standard)
448                             <1>     ; before return from this interrupt
449                             <1>     ; (every function must restore and set
450                             <1>     ; registers except esp, esi, es, ds)
451                             <1>
452                             <1>     ; 29/11/2023 - TRDOS 386 v2.0.7
453 0000185B 0F20DE              <1>     mov     esi, cr3
454 0000185E 56                 <1>     push    esi ; *****
455                             <1>     ;cmp    esi, [k_page_dir]
456                             <1>     ;je     short VBE_func_x
457 0000185F 8B35[80760100]     <1>     mov     esi, [k_page_dir]
458 00001865 0F22DE             <1>     mov     cr3, esi
459                             <1> ;VBE_func_x:
460 00001868 803D[86090000]02   <1>     cmp     byte [vbe3], 2
461 0000186F 7744               <1>     ja      short VESA_VBE3_PMI_CALL ; VBE3 video bios ('PMID')
462                             <1>     ;je     short VBE_func_0  ; Bochs/Qemu/VirtualBox emulator
463 00001871 726C               <1>     jb      short VBE_unknown ; invalid ([vbe3] = 0)
464                             <1>
465                             <1>     ;jmp    VESA_VBE3_PMI_CALL
466                             <1>
467                             <1> VBE_func_0:
468                             <1>     ; Bochs/Plex86 VGAbios VBE extension
469                             <1>     ; (TRDOS 386 v2.0.3 can use VBE graphics modes on emulators)
470                             <1>     ; BOCHS/QEMU/VIRTUALBOX
471                             <1>
472 00001873 8A25[87090000]       <1>     mov     ah, [vbe2bios]
473 00001879 80FCC0             <1>     cmp     ah, 0C0h
474 0000187C 7261               <1>     jb      short VBE_unknown
475 0000187E 80FCC5             <1>     cmp     ah, 0C5h
476 00001881 775C               <1>     ja      short VBE_unknown
477                             <1>
478                             <1>     ; TRDOS 386 is running on BOCHS or QEMU
479 00001883 B44F               <1>     mov     ah, 4Fh
480                             <1> VBE_func_1:
481 00001885 0FB6F0             <1>     movzx   esi, al ; VESA VBE function number
482                             <1>     ;shl    si, 2 ; dword
483                             <1>     ; 02/08/2022
484 00001888 C1E602             <1>     shl     esi, 2
485 0000188B 6683FE14           <1>     cmp     si, N1L
486 0000188F 734E               <1>     jnb     short VBE_unknown
487                             <1>     ;sti
488                             <1>
489 00001891 FF96[A1180000]     <1>     call    dword [esi+N1] ; call VBE function
490                             <1>
491                             <1>     ;jmp    short VBE_bios_return
492                             <1>
493                             <1> VBE_bios_return:
494 00001897 FA                 <1>     cli
495                             <1>     ; 29/11/2023 - TRDOS 386 v2.0.7
496 00001898 5E                 <1>     pop     esi ; *****
497                             <1>     ;cmp    esi, [k_page_dir]
498                             <1>     ;je     short VBE_bios_return_x
499 00001899 0F22DE             <1>     mov     cr3, esi
500                             <1> ;VBE_bios_return_x:
501 0000189C 5E                 <1>     pop     esi ; ****
502 0000189D 5D                 <1>     pop     ebp ; *** ; 27/11/2020
503 0000189E 07                 <1>     pop     es  ; **
504 0000189F 1F                 <1>     pop     ds  ; *
505 000018A0 CF                 <1>     iretd
506                             <1>
507                             <1>     ; 26/11/2020
508                             <1> N1:
509 000018A1 [02190000]         <1>     dd      vbe_biosfn_return_ctrl_info
510 000018A5 [29390000]         <1>     dd      vbe_biosfn_return_mode_info
511 000018A9 [E3390000]         <1>     dd      vbe_biosfn_set_mode
512 000018AD [B33A0000]         <1>     dd      vbe_biosfn_return_current_mode
513 000018B1 [D23A0000]         <1>     dd      vbe_biosfn_save_restore_state
514                             <1>     ;dd     vbe_biosfn_display_window_ctrl
515                             <1>     ;dd     vbe_biosfn_set_get_log_scanline
516                             <1>     ;dd     vbe_biosfn_set_get_disp_start
517                             <1>     ;dd     vbe_biosfn_set_get_dac_pal_frm
518                             <1>     ;dd     vbe_biosfn_set_get_palette_data
519                             <1>
520                             <1> N1L EQU     $ - N1
521                             <1>
522                             <1> VESA_VBE3_PMI_CALL: ; VESA VBE video bios (firmware) functions
523                             <1>                     ; by using VESA VBE3 Protected Mode Interface
524                             <1>
525                             <1>     ; 02/08/2022 - TRDOS 386 v2.0.5
526                             <1>     ; 29/11/2020
527                             <1>     ; 26/11/2020 - TRDOS 386 v2.0.3 video.s
528                             <1>
529                             <1>     ; We are here because..
530                             <1>     ; 'PMID' has been verified by TRDOS 386 v2.0.3 kernel.
531                             <1>     ; (Otherwise bochs/plex86 compatible VBE functions and
532                             <1>     ;  modes would be used on BOCHS/QEMU/VIRTUALBOX emulators
533                             <1>     ;  or only standard/old VGA graphics modes would be used.)
534                             <1>
535                             <1>     ; (TRDOS 386 v2.0.3 can use VESA VBE graphics modes if
536                             <1>     ;  the video bios is full compatible with VBE3 standard)
537                             <1>
538                             <1>     ; 29/11/2020
539                             <1>
540 000018B5 0FB6F0             <1>     movzx   esi, al ; VESA VBE 3 function number
541                             <1>     ;shl    si, 2 ; dword
542                             <1>     ; 02/08/2022
543 000018B8 C1E602             <1>     shl     esi, 2
544 000018BB 6683FE14           <1>     cmp     si, P1L
545 000018BF 731E               <1>     jnb     short VBE_unknown
546                             <1>     ;sti
547                             <1>
```

```
548 000018C1 57                 <1>     push    edi ; *****
549                             <1>
550 000018C2 FF96[CB180000]     <1>     call    dword [esi+P1] ; call VBE 3 function
551                             <1>
552 000018C8 5F                 <1>     pop     edi ; *****
553                             <1>
554 000018C9 EBCC               <1>     jmp     short VBE_bios_return
555                             <1>
556                             <1> P1:
557 000018CB [E5180000]         <1>     dd      vbe3_pmfn_return_ctrl_info
558 000018CF [08190000]         <1>     dd      vbe3_pmfn_return_mode_info
559 000018D3 [44190000]         <1>     dd      vbe3_pmfn_set_mode
560 000018D7 [3F190000]         <1>     dd      vbe3_pmfn_return_current_mode
561 000018DB [4D1A0000]         <1>     dd      vbe3_pmfn_save_restore_state
562                             <1>     ;dd     vbe3_pmfn_display_window_ctrl
563                             <1>     ;dd     vbe3_pmfn_set_get_log_scanline
564                             <1>     ;dd     vbe3_pmfn_set_get_disp_start
565                             <1>     ;dd     vbe3_pmfn_set_get_dac_pal_frm
566                             <1>     ;dd     vbe3_pmfn_set_get_palette_data
567                             <1>     ;dd     vbe3_pmfn_return_pmi ; invalid for TRDOS 386 v2
568                             <1>     ;dd     vbe3_pmfn_set_get_pixel_clock
569                             <1>
570                             <1> P1L EQU     $ - P1
571                             <1>
572                             <1> ;   ; 29/11/2020
573                             <1> ;   mov     edi, VBE3MODEINFOBLOCK >> 4 ; / 16
574                             <1> ;
575                             <1> ;   cmp     al, 04h
576                             <1> ;   jb      short vbe3_pm_f ; function: 4F00h to 4F03h
577                             <1> ;   ja      short vbe3_pmi_f5B ; function: 4F05h to 4F0Bh
578                             <1> ;
579                             <1> ;   ; check buffer length (must be <= 2048 bytes)
580                             <1> ;
581                             <1> ;   and     dl, dl ; 0
582                             <1> ;   jz      short vbe3_pm_f
583                             <1> ;                   ; Return Save/Restore State buffer size
584                             <1> ;
585                             <1> ;   push    ebx  ; buffer address
586                             <1> ;   push    edx  ; function: save (01h) or restore (02h)
587                             <1> ;   call
588                             <1> ;
589                             <1> ;vbe3_pm_f03:
590                             <1> ;   cmp     al, 2
591                             <1> ;   ja      short vbe3_pm_f ; function 4F03h
592                             <1> ;   jb      short vbe3_pm_f1
593                             <1> ;
594                             <1> ;
595                             <1> ;vbe3_pm_f1:
596                             <1> ;
597                             <1> ;
598                             <1> ;vbe3_pmi_f5B:
599                             <1> ;   cmp     al, 09h
600                             <1> ;   jna     short vbe3_pm_f ; funcs 05h to 09h are usable
601                             <1> ;
602                             <1> ;   cmp     al, 0Bh ; Get/Set pixel clock, last function
603                             <1> ;   jne     short VBE_unknown
604                             <1> ;                   ; (do not use 'uncertain' functions
605                             <1> ;                   ; because of system-user buff transfers)
606                             <1> ;vbe3_pm_f:
607                             <1> ;   mov     byte [vbe3_pm_fn], al ; set
608                             <1> ;   ; prepare 16 bit pm segments & registers for pmi call
609                             <1> ;   call    VESA_VBE3_PM_FUNCTION
610                             <1> ;
611                             <1> ;
612                             <1>
613                             <1>     ; 26/11/2020
614                             <1> VBE_unknown:
615 000018DF 66B80001           <1>     mov     ax, 0100h  ; ah = 1 : Function call failed
616                             <1>                        ; al = 0 : Function is not supported
617                             <1>
618 000018E3 EBB2               <1>     jmp     short VBE_bios_return
619                             <1>
620                             <1> vbe3_pmfn_return_ctrl_info:
621                             <1>     ; 02/08/2022 (TRDOS 386 Kernel v2.0.5)
622                             <1>     ; 12/12/2020
623                             <1>     ;
624                             <1>     ; VBE function 4F00h - Return VBE Controller Information
625                             <1>     ;
626                             <1>     ; Input:
627                             <1>     ;      EDI = Pointer to buffer in which to place
628                             <1>     ;            VbeInfoBlock structure
629                             <1>     ;
630                             <1>     ;      AX = 4F00h
631                             <1>     ; Output:
632                             <1>     ;      AX = VBE return status
633                             <1>     ;            AX = 004Fh -> succeeded
634                             <1>     ;            AX <> 004Fh -> failed
635                             <1>     ;
636                             <1>     ; Modified registers: eax (+ edi for kernel's own call)
637                             <1>
638                             <1>     ; NOTE: TRDOS 386 v2 (v2.0.3) kernel calls this function
639                             <1>     ; during startup while cpu is in real mode
640                             <1>     ; (by using int 10h, 4F02h) and saves VbeInfoBlock at
641                             <1>     ; VBE3INFOBLOCK address (97E00h for TRDOS 386 v2.0.3).
642                             <1>     ;
643                             <1>     ; So...
644                             <1>     ; This VBE function is adjusted to return/move same info
645                             <1>     ; from VBE3INFOBLOCK to user's buffer in EDI.
646                             <1>
647                             <1>     ; int 31h (int 10h) entrance
648                             <1>
649 000018E5 21FF               <1>     and     edi, edi
650 000018E7 7423               <1>     jz      short vbe3_func_fail ; invalid buffer address !
651                             <1>
652                             <1> ;_vbe3_pmfn_return_ctrl_info:
653                             <1>     ;or     edi, edi
654                             <1>     ;jnz    short _vbe_biosfn_return_ctrl_info
655                             <1>     ;
656                             <1>     ;; this option may not be necessary - 12/12/2020
657                             <1>     ;
658                             <1>     ;; edi = 0, kernel forces to get ctrl info again
659                             <1>     ;;         by using VESA VBE3 video bios's pmi
660                             <1>     ;
661                             <1>     ;;push  edi
662                             <1>     ;; far call to VESA VBE3 PMI
663                             <1>     ;;mov   ax, 4F00h ; Return VBE Controller Info
664                             <1>     ;mov    edi, VBE3INFOBLOCK-VBE3SAVERESTOREBLOCK
665                             <1>     ;; ES selector base address = VBE3SAVERESTOREBLOCK
666                             <1>     ;call   int10h_32bit_pmi
667                             <1>     ;;pop   edi
668                             <1>     ;mov    edi, VBE3INFOBLOCK ; retn to the kernel sub
669                             <1>     ;cmp    ax, 004Fh
670                             <1>     ;je     short vbe_ctrl_info_retn
671                             <1>     ;stc
```

```
                            <1>     ;retn
                            <1>
                            <1> _vbe_biosfn_return_ctrl_info:
000018E9 57                 <1>     push    edi
000018EA 51                 <1>     push    ecx
000018EB BE007E0900         <1>     mov     esi, VBE3INFOBLOCK
                            <1>     ;mov    ecx, 512
                            <1>     ; 02/08/2022
000018F0 29C9               <1>     sub     ecx, ecx
000018F2 B502               <1>     mov     ch, 2
                            <1>     ; ecx = 512
000018F4 E8F1F10000         <1>     call    transfer_to_user_buffer
000018F9 59                 <1>     pop     ecx
000018FA 5F                 <1>     pop     edi
000018FB 720F               <1>     jc      short vbe3_func_fail
                            <1>
000018FD 31C0               <1>     xor     eax, eax
000018FF B04F               <1>     mov     al, 4Fh ; successful
                            <1> ;vbe_ctrl_info_retn:
00001901 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>
00001902 21FF               <1>     and     edi, edi
00001904 7406               <1>     jz      short vbe3_func_fail ; invalid buffer addr !
00001906 EBE1               <1>     jmp     short _vbe_biosfn_return_ctrl_info
                            <1>
                            <1> vbe3_pmfn_return_mode_info:
                            <1>     ; 02/08/2022 (TRDOS 386 Kernel v2.0.5)
                            <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>
00001908 09FF               <1>     or      edi, edi
0000190A 7506               <1>     jnz     short _vbe3_pmfn_return_mode_info
                            <1>
                            <1> vbe3_func_fail:
0000190C B84F010000         <1>     mov     eax, 014Fh ; ah = 1 : Function call failed
                            <1>                        ; al = 4Fh : Function is supported
00001911 C3                 <1>     retn
                            <1>
                            <1>     ; jump from '_vbe_biosfn_return_mode_info'
                            <1> _vbe3_pmfn_return_mode_info:
00001912 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
00001913 BF00060000         <1>     mov     edi, VBE3MODEINFOBLOCK-VBE3SAVERESTOREBLOCK
                            <1>     ; ES selector base address = VBE3SAVERESTOREBLOCK
00001918 E812010000         <1>     call    int10h_32bit_pmi
                            <1>
0000191D 5F                 <1>     pop     edi
                            <1>
                            <1>     ;cmp    ax, 004Fh
                            <1>     ;jne    short vbe3_func_retn  ; failed !
                            <1>
0000191E 21FF               <1>     and     edi, edi
                            <1>     ;jz     short vbe3_func_success
                            <1>     ; 21/12/2020
00001920 741C               <1>     jz      short vbe3_func_retn
                            <1>
00001922 6683F84F           <1>     cmp     ax, 004Fh
00001926 7516               <1>     jne     short vbe3_func_retn  ; failed !
                            <1>
00001928 51                 <1>     push    ecx
00001929 BE007C0900         <1>     mov     esi, VBE3MODEINFOBLOCK
                            <1>     ;mov    ecx, 256
                            <1>     ; 02/08/2022
0000192E 29C9               <1>     sub     ecx, ecx
00001930 FEC5               <1>     inc     ch
                            <1>     ; ecx = 256
00001932 E8B3F10000         <1>     call    transfer_to_user_buffer
00001937 59                 <1>     pop     ecx
00001938 72D2               <1>     jc      short vbe3_func_fail
                            <1>
0000193A 31C0               <1>     xor     eax, eax
0000193C B04F               <1>     mov     al, 4Fh ; successful
                            <1> vbe3_func_success:
                            <1> vbe3_func_retn:
0000193E 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
0000193F E9EB000000             <1>     jmp    int10h_32bit_pmi
                                <1>
                                <1> vbe3_pmfn_set_mode:
                                <1>     ; 02/08/2022 (TRDOS 386 Kernel v2.0.5)
                                <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
00001944 57                     <1>     push   edi ; *****
00001945 F6C708                 <1>     test   bh, 8  ; Use user specified CRTC values
00001948 7530                   <1>     jnz    short vbe3_sm_3  ; for refresh rate
                                <1> vbe3_sm_4:
0000194A 80E7C1                 <1>     and    bh, 0C1h ; use bit 15, 14, 8 only (for bh)
                                <1>     ;mov   [vbe_mode_x], bh
                                <1>
0000194D 803D[1E680000]03       <1>     cmp    byte [CRT_MODE], 3 ; is current mode 03h ?
00001954 7509                   <1>     jne    short vbe3_sm_1    ; no
                                <1>
                                <1>     ; save mode 03h video pages and cursor positions
00001956 57                     <1>     push   edi ; **!***
00001957 51                     <1>     push   ecx ; ******
                                <1>     ;push  esi
                                <1>
00001958 E8CE040000             <1>     call   save_mode3_multiscreen
                                <1>
                                <1>     ;pop   esi
0000195D 59                     <1>     pop    ecx ; ******
0000195E 5F                     <1>     pop    edi ; **!***
                                <1> vbe3_sm_1:
                                <1>     ; ax = 4F02h
                                <1>     ; bx = video mode number (vbe2 type)
0000195F E8CB000000             <1>     call   int10h_32bit_pmi
                                <1>                    ; call to far call to VBE3 PMI
                                <1>
00001964 6683F84F               <1>     cmp    ax, 004Fh ; succeeded ?
00001968 750E                   <1>     jne    short vbe3_sm_2
                                <1>     ; set current mode byte/sign to extended (SVGA) mode
0000196A C605[1E680000]FF       <1>     mov    byte [CRT_MODE], 0FFh ; VESA VBE mode
                                <1>     ; set current VBE mode word to bx input
00001971 66891D[EE9C0100]       <1>     mov    [video_mode], bx
                                <1> vbe3_sm_2:
                                <1>     ; 22/12/2020
00001978 5F                     <1>     pop    edi ; *****
00001979 C3                     <1>     retn
                                <1>
```

```
                                <1> vbe3_sm_3:
                                <1>     ; 22/12/2020
                                <1>     ; copy user's CRTCInfoBlock to the buffer
0000197A 51                     <1>     push    ecx
0000197B 89FE                   <1>     mov     esi, edi
0000197D BF807D0900             <1>     mov     edi, VBE3CRTCINFOBLOCK
                                <1>     ;mov    ecx, 64
                                <1>     ; 02/08/2022
00001982 29C9                   <1>     sub     ecx, ecx
00001984 B140                   <1>     mov     cl, 64
00001986 E8A9F10000             <1>     call    transfer_from_user_buffer
0000198B 59                     <1>     pop     ecx
                                <1>     ; set offset (es base addr is VBE3SAVERESTOREBLOCK)
0000198C 81EF00760900           <1>     sub     edi, VBE3SAVERESTOREBLOCK
00001992 EBB6                   <1>     jmp     short vbe3_sm_4
                                <1>
                                <1> vesa_vbe3_pmi:
                                <1>     ; 29/11/2023 - TRDOS 386 v2.0.7
                                <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 parameters before and after)
                                <1>
                                <1>     ; 29/11/2023 - TRDOS 386 v2.0.7
00001994 50                     <1>     push    eax
00001995 0F20D8                 <1>     mov     eax, cr3
00001998 870424                 <1>     xchg    eax, [esp] ; **!**
0000199B 50                     <1>     push    eax
                                <1>     ;cmp    esi, [k_page_dir]
                                <1>     ;je     short vesa_vbe3_pmi_x
0000199C A1[80760100]           <1>     mov     eax, [k_page_dir]
000019A1 0F22D8                 <1>     mov     cr3, eax
                                <1> ;vesa_vbe3_pmi_x:
000019A4 58                     <1>     pop     eax
                                <1>
                                <1>     ; 06/12/2020
000019A5 20E4                   <1>     and     ah, ah  ; 0 = set mode function
000019A7 7402                   <1>     jz      short vbe3_pmi_0
000019A9 EB7D                   <1>     jmp     vbe3_pmi_9
                                <1>
                                <1> vbe3_pmi_0:
                                <1>     ; 07/12/2020
000019AB 88C4                   <1>     mov     ah, al
000019AD 80E480                 <1>     and     ah, 80h ; 0 or 80h
000019B0 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
000019B2 8825[17830100]         <1>     mov     byte [noclearmem], ah ; 0 or 80h
                                <1>     ; check current video mode if it is 03h
000019B8 803D[1E680000]03       <1>     cmp     byte [CRT_MODE], 3 ; current mode
000019BF 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
000019C1 56                     <1>     push    esi
000019C2 57                     <1>     push    edi
000019C3 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
000019C4 E862040000             <1>     call    save_mode3_multiscreen
                                <1>
000019C9 59                     <1>     pop     ecx
000019CA 5F                     <1>     pop     edi
000019CB 5E                     <1>     pop     esi
                                <1> vbe3_pmi_3:
                                <1>     ; 08/12/2020
                                <1>     ; 07/12/2020
                                <1>     ; case 3 or case 4
000019CC A2[1E680000]           <1>     mov     [CRT_MODE], al
000019D1 3C03                   <1>     cmp     al, 3
000019D3 7407                   <1>     je      short vbe3_pmi_4
                                <1>     ; case 4:
                                <1>     ; Current mode is not 03h and also new mode is not 03h
000019D5 800D[15830100]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>
000019DC E84E000000             <1>     call    int10h_32bit_pmi
                                <1>
000019E1 803D[1E680000]03       <1>     cmp     byte [CRT_MODE], 3  ; new video mode
                                <1>     ;jne    vbe3_pmi_8      ; video mode <> 03h
000019E8 7532                   <1>     jne     short vbe3_pmi_8
                                <1>
                                <1>     ;push   eax ; 04/12/2020
000019EA 53                     <1>     push    ebx
000019EB 51                     <1>     push    ecx
000019EC 52                     <1>     push    edx
000019ED 57                     <1>     push    edi ; 03/12/2020
                                <1>
                                <1>     ; 12/12/2020
000019EE 56                     <1>     push    esi
000019EF E869040000             <1>     call    restore_mode3_multiscreen
000019F4 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>     ;
                                <1>     ;mov    cl, al
                                <1> ;vbe3_pmi_5:
                                <1>     ;add    word [CRT_START], 4096
                                <1>     ;dec    cl
                                <1>     ;jnz    short vbe3_pmi_5
                                <1>
000019F5 B405                   <1>     mov     ah, 05h ; set current video page
                                <1>     ; al = video page
000019F7 E833000000             <1>     call    int10h_32bit_pmi
                                <1>
                                <1>     ; check current cursor position & set it again if not 0,0
                                <1>     ;movzx  ebx, byte [ACTIVE_PAGE]
000019FC 0FB6D8                 <1>     movzx   ebx, al
000019FF D0E3                   <1>     shl     bl, 1
00001A01 81C3[9E760100]         <1>     add     ebx, CURSOR_POSN
00001A07 668B13                 <1>     mov     dx, [ebx]
00001A0A 6621D2                 <1>     and     dx, dx
00001A0D 7409                   <1>     jz      short vbe3_pmi_7
                                <1>
                                <1>     ;dx = cursor position (dl = column, dh = row)
                                <1>     ;mov    bh, [ACTIVE_PAGE] ; 06/12/2020
00001A0F 88C7                   <1>     mov     bh, al
00001A11 B402                   <1>     mov     ah, 02h ; set cursor position
00001A13 E817000000             <1>     call    int10h_32bit_pmi
                                <1>
                                <1>     ;jmp    short vbe3_pmi_7
                                <1>
                                <1> ;vbe3_pmi_6:
                                <1> ;   ; 07/12/2020
                                <1> ;   ; case 1, previous mode is 03h, current mode is 03h
                                <1> ;   ; 03/12/2020
                                <1> ;   cmp     byte [noclearmem], 0
                                <1> ;   jna     short vbe3_pmi_7 ; do not clear memory
                                <1> ;   ; clear video page
                                <1> ;   mov     ecx, 1024 ; 4096/4
                                <1> ;   mov     eax, 07200720h
                                <1> ;   mov     edi, 0B8000h ; [crt_base]
                                <1> ;   add     di, [CRT_START]
                                <1> ;   rep     stosd   ; FILL THE REGEN BUFFER WITH BLANKS
                                <1>
                                <1> vbe3_pmi_7:
00001A18 5F                     <1>     pop     edi
00001A19 5A                     <1>     pop     edx
00001A1A 59                     <1>     pop     ecx
00001A1B 5B                     <1>     pop     ebx
                                <1>     ;pop    eax ; 04/12/2020
                                <1> vbe3_pmi_8:
                                <1>     ; 04/12/2020
                                <1>     ;(TRDOS 386 v2.0.3, INT 31h, ah=0 return)
00001A1C 31C0                   <1>     xor     eax, eax  ; eax = 0 -> succesful
                                <1> vesa_vbe3_pmi_retn:
                                <1>     ; 29/11/2023 - TRDOS 386 v2.0.7
00001A1E 870424                 <1>     xchg    eax, [esp] ; **!**
                                <1>     ;cmp    eax, [k_page_dir]
                                <1>     ;je     short vesa_vbe3_pmi_retn_x
00001A21 0F22D8                 <1>     mov     cr3, eax
                                <1> ;vesa_vbe3_pmi_retn_x:
00001A24 58                     <1>     pop     eax
```

```
                                <1>     ;
00001A25 07                     <1>     pop     es  ; **
00001A26 1F                     <1>     pop     ds  ; *
00001A27 CF                     <1>     iretd
                                <1>
                                <1> vbe3_pmi_9:
                                <1>     ; 06/12/2020
                                <1>     ;cmp    ah, 10h ; Set/Get Palette Registers
                                <1>     ;jnb    short vbe3_pmi_10
                                <1>     ; 05/12/2020
00001A28 E802000000             <1>     call    int10h_32bit_pmi
00001A2D EBEF                   <1>     jmp     short vesa_vbe3_pmi_retn
                                <1>
                                <1> ;vbe3_pmi_10:
                                <1>     ; 06/12/2020
                                <1>     ;jmp    VGA_funcs_0
                                <1>
                                <1> int10h_32bit_pmi:
                                <1>     ; 03/12/2020
                                <1>     ; 28/11/2020
                                <1>     ; calling standard VGA Bios (INT 10h) functions
                                <1>     ; by using 32 bit protected mode interface of
                                <1>     ; VESA VBE3 Video Bios (with 'PMID' signature)
                                <1>
                                <1>     ; 03/12/2020
                                <1>     ; eax, ebx, ecx, edx, edi will be used by vbios pmi
                                <1>     ; (esi and ebp will not be used)
                                <1>
                                <1>     ; 03/12/2020
00001A2F 56                     <1>     push    esi
00001A30 C1E010                 <1>     shl     eax, 16  ; move function number (ax) to hw
00001A33 8B35[F49C0100]         <1>     mov     esi, [pmid_addr] ; linear address of
                                <1>                              ; PMInfo.Entrypoint pointer
                                <1>     ;mov    ax, [esi+PMInfo.EntryPoint]
00001A39 668B06                 <1>     mov     ax, [esi]
00001A3C C1C010                 <1>     rol     eax, 16 ; move PM entry address to hw
                                <1>                     ; and move function number to lw (ax)
00001A3F 5E                     <1>     pop     esi
                                <1>
                                <1>     ; top of stack: ; (*)
                                <1>     ; return (the caller) address of "int10h_32bit_pmi"
                                <1>
00001A40 E91FEDFFFF             <1>     jmp     _VBE3PMI_fcall ; will return to the caller (*)
                                <1>
                                <1> _vbe3_pmfn_srs_8:
                                <1>     ; 17/01/2021
00001A45 31DB                   <1>     xor     ebx, ebx ; points to VBE3SAVERESTOREBLOCK
                                <1> _vbe3_pmfn_srs_9: ; 24/01/2021
00001A47 66B8044F               <1>     mov     ax, 4F04h
00001A4B EBE2                   <1>     jmp     short int10h_32bit_pmi
                                <1>
                                <1> vbe3_pmfn_save_restore_state:
                                <1>     ; 02/08/2022 (TRDOS 386 Kernel v2.0.5)
                                <1>     ; 24/01/2021
                                <1>     ; 23/01/2021
                                <1>     ; 16/01/2021 - 17/01/2021
                                <1>     ; 14/01/2021
                                <1>     ;
                                <1>     ; VBE function 4F04h - Save/Restore Video State
                                <1>     ;
                                <1>     ; Input:
                                <1>     ;       DL = sub function
                                <1>     ;       CL = requested state
                                <1>     ;      EBX = pointer to buffer (if DL<>00h)
                                <1>     ;       AX = 4F04h
                                <1>     ; Output:
                                <1>     ;       AX = 004Fh (successful)
                                <1>     ;       AH > 0 -> error
                                <1>     ;       BX = Number of 64-byte blocks
                                <1>     ;            to hold the state buffer (if DL=00h)
                                <1>
                                <1>     ; Modified registers: eax, ebx, esi, edi
                                <1>
00001A4D 21DB                   <1>     and     ebx, ebx ; user's buffer address
00001A4F 750A                   <1>     jnz     short _vbe3_pmfn_save_restore_state
                                <1>
00001A51 08D2                   <1>     or      dl, dl
00001A53 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:
00001A55 B84F010000             <1>     mov     eax, 014Fh ; ah = 1 : Function call failed
                                <1>                        ; al = 4Fh : Function is supported
                                <1> _vbe3_srs_retn:
00001A5A C3                     <1>     retn
                                <1>
                                <1> _vbe3_pmfn_save_restore_state:
00001A5B 20D2                   <1>     and     dl, dl
00001A5D 7555                   <1>     jnz     short _vbe3_pmfn_srs_2
                                <1> _vbe3_pmfn_srs:
00001A5F 31DB                   <1>     xor     ebx, ebx
                                <1> _vbe3_pmfn_srs_0:
                                <1>     ; 24/01/2021
00001A61 83F90F                 <1>     cmp     ecx, 0Fh
                                <1>     ;ja     short _vbe3_pmfn_srs_1
00001A64 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)
00001A66 80E1FD                 <1>     and     cl, ~2 ; and cl, not 2
                                <1>
                                <1>     ; 24/01/2021
                                <1>     ;mov    bl, 1
00001A69 FEC3                   <1>     inc     bl ; = 1
                                <1>     ;shl    bx, cl
                                <1>     ; 02/08/2022
00001A6B D3E3                   <1>     shl     ebx, cl
00001A6D 66231D[F89C0100]       <1>     and     bx, [vbe3stbsflags]
00001A74 7415                   <1>     jz      short _vbe3_pmfn_srs_1
                                <1>     ;mov    bx, cx
00001A76 89CB                   <1>     mov     ebx, ecx ; <= 15
00001A78 D0E3                   <1>     shl     bl, 1 ; 0, 2, 8 .. 30
00001A7A 668B9B[833B0000]       <1>     mov     bx, [vbestatebufsize+ebx]
00001A81 89DF                   <1>     mov     edi, ebx
                                <1>     ; edi = state buffer size in bytes
                                <1>     ;shr    bx, 6 ; / 64
                                <1>     ; 02/08/2022
```

```
1292 00001A83 C1EB06              <1>     shr     ebx, 6
1293                              <1>     ;mov    ax, 4Fh
1294 00001A86 28E4                <1>     sub     ah, ah
1295 00001A88 B04F                <1>     mov     al, 4Fh
1296 00001A8A C3                  <1>     retn
1297                              <1> _vbe3_pmfn_srs_1:
1298                              <1>     ; ax = 4F04h
1299                              <1>     ;call   int10h_32bit_pmi
1300                              <1>     ; 24/01/2021
1301                              <1>     ;call   _vbe3_pmfn_srs_8
1302                              <1>     ; ebx = 0
1303 00001A8B E8B7FFFFFF          <1>     call    _vbe3_pmfn_srs_9
1304 00001A90 6683F84F            <1>     cmp     ax, 004Fh
1305 00001A94 75C4                <1>     jne     short _vbe3_srs_retn
1306                              <1>     ; 24/01/2021
1307                              <1>     ;cmp    ecx, 0Fh
1308                              <1>     ;ja     short _vbe3_srs_retn
1309                              <1>     ; 24/01/2021
1310                              <1>     ;mov    ax, 1
1311 00001A96 B001                <1>     mov     al, 1
1312                              <1>     ;shl    ax, cl
1313                              <1>     ; 02/08/2022
1314 00001A98 D3E0                <1>     shl     eax, cl
1315 00001A9A 660905[F89C0100]    <1>     or      [vbe3stbsflags], ax ; set flag for state option
1316                              <1>     ; 23/01/2021
1317 00001AA1 89DF                <1>     mov     edi, ebx
1318 00001AA3 89C8                <1>     mov     eax, ecx
1319 00001AA5 D0E0                <1>     shl     al, 1
1320                              <1>     ;shl    di, 6 ; * 64
1321                              <1>     ; 02/08/2022
1322 00001AA7 C1E706              <1>     shl     edi, 6
1323 00001AAA 6689B8[833B0000]    <1>     mov     [vbestatebufsize+eax], di
1324                              <1>                             ; save buf size for option
1325                              <1>     ;xchg   edi, ebx
1326                              <1>     ; edi = state buffer size in bytes
1327 00001AB1 B04F                <1>     mov     al, 4Fh
1328 00001AB3 C3                  <1>     retn
1329                              <1>
1330                              <1> _vbe3_pmfn_srs_2:
1331                              <1>     ; 24/01/2021
1332                              <1>     ; !!! CLEAR CL BIT 2 !!!
1333                              <1>     ; (when bit 2 is set, function causes cpu exception)
1334                              <1>     ; BIOS data will not be saved and restored
1335                              <1>     ; (to prevent protected mode page fault error)
1336                              <1>
1337 00001AB4 F6C1FD              <1>     test    cl, ~2 ; test cl, not 2
1338 00001AB7 749C                <1>     jz      short _vbe3_pmfn_srs_fail
1339                              <1>
1340 00001AB9 80FA02              <1>     cmp     dl, 2
1341 00001ABC 7748                <1>     ja      short _vbe3_pmfn_srs_5
1342                              <1>
1343                              <1>     ;and    cl, ~2 ; and cl, not 2
1344                              <1>
1345 00001ABE 53                  <1>     push    ebx ; * ; buffer address
1346                              <1>     ; save or restore state
1347                              <1>     ; (get required buffer size at first)
1348 00001ABF 52                  <1>     push    edx ; **
1349 00001AC0 28D2                <1>     sub     dl, dl ; 0
1350 00001AC2 E898FFFFFF          <1>     call    _vbe3_pmfn_srs
1351 00001AC7 5A                  <1>     pop     edx ; **
1352                              <1>     ; 24/01/2021
1353 00001AC8 5B                  <1>     pop     ebx ; *
1354 00001AC9 08E4                <1>     or      ah, ah
1355 00001ACB 7538                <1>     jnz     short _vbe3_pmfn_srs_4 ; error
1356                              <1>
1357                              <1>     ; edi = buffer size in bytes
1358 00001ACD 81FF00080000        <1>     cmp     edi, 2048
1359 00001AD3 772B                <1>     ja      short _vbe3_pmfn_srs_3
1360                              <1>
1361 00001AD5 80FA01              <1>     cmp     dl, 1
1362 00001AD8 7531                <1>     jne     short _vbe3_pmfn_srs_6 ; restore state
1363                              <1>
1364                              <1>     ; save video state
1365                              <1>     ;xor    ebx, ebx ; points to VBE3SAVERESTOREBLOCK
1366                              <1>     ;mov    ax, 4F04h
1367                              <1>     ;call   int10h_32bit_pmi
1368                              <1>
1369                              <1>     ; 24/01/2021
1370 00001ADA E842000000          <1>     call    _vbe3_pmfn_srs_7
1371                              <1>
1372 00001ADF 6683F84F            <1>     cmp     ax, 004Fh
1373 00001AE3 7520                <1>     jne     short _vbe3_pmfn_srs_4
1374                              <1>
1375 00001AE5 09DB                <1>     or      ebx, ebx  ; kernel ('sysvideo') ?
1376 00001AE7 741C                <1>     jz      short _vbe3_pmfn_srs_4 ; yes
1377                              <1>
1378                              <1>     ; the caller is user
1379 00001AE9 51                  <1>     push    ecx ; *
1380 00001AEA 89F9                <1>     mov     ecx, edi ; state buffer size
1381 00001AEC BE00760900          <1>     mov     esi, VBE3SAVERESTOREBLOCK ; source
1382                              <1>                                       ; (vbe3 pmi buff)
1383 00001AF1 89DF                <1>     mov     edi, ebx       ; destination (user buff)
1384 00001AF3 E8F2EF0000          <1>     call    transfer_to_user_buffer
1385 00001AF8 59                  <1>     pop     ecx ; *
1386 00001AF9 7205                <1>     jc      short _vbe3_pmfn_srs_3
1387                              <1>
1388 00001AFB 29C0                <1>     sub     eax, eax
1389 00001AFD B04F                <1>     mov     al, 4Fh
1390 00001AFF C3                  <1>     retn
1391                              <1>
1392                              <1>     ; 24/01/2021
1393                              <1> _vbe3_pmfn_srs_3:
1394 00001B00 B84F010000          <1>     mov     eax, 014Fh
1395                              <1> _vbe3_pmfn_srs_4:
1396 00001B05 C3                  <1>     retn
1397                              <1> _vbe3_pmfn_srs_5:
1398 00001B06 31C0                <1>     xor     eax, eax
1399 00001B08 FEC4                <1>     inc     ah
1400                              <1>     ; eax = 0100h, function is not supported
1401 00001B0A C3                  <1>     retn
1402                              <1>
1403                              <1> _vbe3_pmfn_srs_6:
1404                              <1>     ; restore video state
1405                              <1>     ; 24/01/2021
1406                              <1>     ;pop    ebx ; *
1407                              <1>     ; 23/01/2021
1408 00001B0B 09DB                <1>     or      ebx, ebx ; 0 ?
1409 00001B0D 7412                <1>     jz      short _vbe3_pmfn_srs_7 ; 'sysvideo' call
1410                              <1>     ; 24/01/2021
1411                              <1>     ;jz     _vbe3_pmfn_srs_8
1412 00001B0F 89DE                <1>     mov     esi, ebx
1413                              <1>     ; esi = user's video state buffer
1414 00001B11 51                  <1>     push    ecx ; *
1415 00001B12 89F9                <1>     mov     ecx, edi ; state buffer size
```

```
1416 00001B14 BF00760900          <1>     mov     edi, VBE3SAVERESTOREBLOCK ; destination
1417                              <1>                                       ; (vbe3 pmi buff)
1418                              <1>     ;mov    esi, ebx        ; source (user buff)
1419 00001B19 E816F00000          <1>     call    transfer_from_user_buffer
1420 00001B1E 59                  <1>     pop     ecx ; *
1421 00001B1F 72DF                <1>     jc      short _vbe3_pmfn_srs_3
1422                              <1> _vbe3_pmfn_srs_7:
1423 00001B21 53                  <1>     push    ebx ; *
1424                              <1>     ; restore video state
1425                              <1>     ;xor    ebx, ebx ; points to VBE3SAVERESTOREBLOCK
1426                              <1>     ;mov    ax, 4F04h
1427                              <1>     ;call   int10h_32bit_pmi
1428                              <1>     ; 17/01/2021
1429 00001B22 E81EFFFFFF          <1>     call    _vbe3_pmfn_srs_8
1430 00001B27 5B                  <1>     pop     ebx ; *
1431 00001B28 C3                  <1>     retn
1432                              <1>
1433                              <1> VIDEO_STATE:
1434                              <1>     ; 26/06/2016
1435                              <1>     ; 12/05/2016
1436                              <1>     ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
1437                              <1>
1438                              <1> ;-------------------------------------------------
1439                              <1> ; VIDEO STATE
1440                              <1> ;  RETURNS THE CURRENT VIDEO STATE IN AX
1441                              <1> ;  AH = NUMBER OF COLUMNS ON THE SCREEN
1442                              <1> ;  AL = CURRENT VIDEO MODE
1443                              <1> ;  BH = CURRENT ACTIVE PAGE
1444                              <1> ;-------------------------------------------------
1445                              <1>
1446 00001B29 8A25[20680000]      <1>     mov     ah, [CRT_COLS] ; GET NUMBER OF COLUMNS
1447 00001B2F A0[1E680000]        <1>     mov     al, [CRT_MODE] ; CURRENT MODE
1448                              <1>     ;movzx  esi, al
1449                              <1>     ;mov    ah, [esi+M6]
1450                              <1>     ; BH = active page
1451 00001B34 8A3D[AE760100]      <1>     mov     bh, [ACTIVE_PAGE] ; GET CURRENT ACTIVE PAGE
1452 00001B3A FA                  <1>     cli     ; 02/01/2017
1453 00001B3B 5D                  <1>     pop     ebp             ; RECOVER REGISTERS
1454 00001B3C 5F                  <1>     pop     edi
1455 00001B3D 5E                  <1>     pop     esi
1456 00001B3E 59                  <1>     pop     ecx             ; DISCARD SAVED BX
1457 00001B3F EB41                <1>     jmp     short M15       ; RETURN TO CALLER
1458                              <1>
1459                              <1> set_mode_ncm:
1460                              <1>     ; 17/11/2020 (TRDOS 386 v2.0.3)
1461                              <1>     ; 04/07/2016 - TRDOS 386 (TRDOS v2.0)
1462                              <1>     ; set mode without clearing the video memory
1463                              <1>     ; (ony for graphics modes)
1464                              <1>
1465                              <1>     ;cmp    al, 7 ; IBM PC CGA modes
1466                              <1>     ;jna    short SET_MODE ; normal function (clear)
1467                              <1>     ;; do not clear memory
1468                              <1>     ;;mov   [noclearmem], al ; > 0
1469                              <1>     ;mov    byte [noclearmem], 80h ; 17/11/2020
1470                              <1>     ;call   _set_mode
1471                              <1>     ;mov    byte [noclearmem], 0
1472                              <1>         ;jmp     short VIDEO_RETURN
1473                              <1>
1474                              <1>     ; 17/11/2020 (TRDOS v2.0.3)
1475 00001B41 0C80                <1>     or      al, 80h ; not clear memory option
1476                              <1>
1477                              <1>     ; 05/12/2020
1478                              <1>     ; 27/11/2020
1479                              <1>     ; 17/11/2020
1480                              <1>     ; 08/08/2016, 10/08/2016
1481                              <1>     ; 29/07/2016, 30/07/2016
1482                              <1>     ; 25/07/2016, 26/07/2016, 27/07/2016
1483                              <1>     ; 02/07/2016, 18/07/2016, 23/07/2016
1484                              <1>     ; 24/06/2016, 26/06/2016
1485                              <1>     ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
1486                              <1> SET_MODE:
1487                              <1>     ; For 32 bit TRDOS and Retro UNIX 386:
1488                              <1>     ;       valid video mode: 03h only!
1489                              <1>     ;       (VGA modes will be selected with another routine)
1490                              <1>     ;
1491                              <1>     ; set_txt_mode ; 80*25 (16 fore colors, 8 back colors)
1492                              <1>
1493                              <1>     ; 27/11/2020
1494                              <1>
1495                              <1>     ; Check if current mode is
1496                              <1>     ; Bochs/Plex86 VBE graphics mode
1497 00001B43 803D[1E680000]FF     <1>     cmp     byte [CRT_MODE], 0FFh ; VESA VBE graphics mode
1498 00001B4A 7220                <1>     jb      short _set_mode_      ; signature
1499                              <1>                                   ; VBE mode number is in
1500                              <1>                                   ; [video_mode] bit 0to8
1501 00001B4C 88C3                <1>     mov     bl, al ; save video mode
1502 00001B4E E87C230000          <1>     call    dispi_get_enable
1503 00001B53 50                  <1>     push    eax ; save current VBE dispi status
1504                              <1>     ; Disable Bochs/Plex86 VBE dispi
1505                              <1>     ;mov    ax, 0 ; VBE_DISPI_DISABLED
1506 00001B54 31C0                <1>     xor     eax, eax ; 0
1507 00001B56 E849230000          <1>     call    dispi_set_enable
1508 00001B5B 88D8                <1>     mov     al, bl ; restore video mode
1509 00001B5D E827000000          <1>     call    _set_mode
1510 00001B62 58                  <1>     pop     eax ; restore current VBE dispi status
1511 00001B63 7313                <1>     jnc     short VIDEO_RETURN
1512                              <1>     ; ! unimplemented or invalid video mode number !
1513                              <1>     ; VBE dispi must be enabled again
1514                              <1>     ; (return to run on current VBE graphics mode)
1515                              <1>     ;;mov   al, [video_mode+1] ; bit 8 to 15
1516                              <1>     ;;and   al, 0C0h ; isolate bit 14 and bit 15
1517                              <1>     ;;or    al, 1 ; VBE_DISPI_ENABLED
1518 00001B65 E83A230000          <1>     call    dispi_set_enable
1519 00001B6A EB07                <1>     jmp     short _video_func_err
1520                              <1>
1521                              <1> _set_mode_:
1522                              <1>     ; VGA bios (non-VBE) 'setmode' procedure
1523                              <1>
1524                              <1>     ; 26/11/2020 (TRDOS v2.0.3)
1525                              <1>
1526                              <1> ;-------------------------------------------------
1527                              <1> ; SET MODE                                      :
1528                              <1> ;   THIS ROUTINE INITIALIZES THE ATTACHMENT TO    :
1529                              <1> ;   THE SELECTED MODE, THE SCREEN IS BLANKED.     :
1530                              <1> ; INPUT                                           :
1531                              <1> ;   (AL) - MODE SELECTED (RANGE 0-7)           :
1532                              <1> ; OUTPUT                                        :
1533                              <1> ;   NONE                                        :
1534                              <1> ;-------------------------------------------------
1535                              <1>
1536 00001B6C E818000000          <1>     call    _set_mode ; 24/06/2016 (set_txt_mode)
1537                              <1>     ; 26/11/2020
1538 00001B71 7305                <1>     jnc     short VIDEO_RETURN
1539                              <1>
```

```
1540                                    <1>     ; 26/11/2020
1541                                    <1> _video_func_err:
1542 00001B73 31C0                     <1>     xor     eax, eax ; function call failed
1543 00001B75 48                       <1>     dec     eax  ; 0FFFFFFFFh ; - 1
1544 00001B76 EB05                     <1>     jmp     short _video_return
1545                                    <1>
1546                                    <1> ; 12/05/2016
1547                                    <1> ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
1548                                    <1>
1549                                    <1> ;-----      NORMAL RETURN FROM ALL VIDEO RETURNS
1550                                    <1>
1551                                    <1> VIDEO_RETURN:
1552 00001B78 A1[08830100]             <1>     mov     eax, [video_eax] ; 12/05/2016
1553                                    <1> _video_return:
1554 00001B7D FA                       <1>     cli ; 02/01/2017
1555 00001B7E 5D                       <1>     pop     ebp ; ******** ; 26/11/2020
1556 00001B7F 5F                       <1>     pop     edi ; *******
1557 00001B80 5E                       <1>     pop     esi ; ******
1558 00001B81 5B                       <1>     pop     ebx ; *****
1559                                    <1> M15:; VIDEO_RETURN_C
1560                                    <1>     ;;15/01/2017
1561                                    <1>     ; 02/01/2017
1562                                    <1>     ;;mov  byte [intflg], 0
1563                                    <1>     ;
1564 00001B82 59                       <1>     pop     ecx ; **** ; 26/11/2020
1565 00001B83 5A                       <1>     pop     edx ; ***
1566 00001B84 1F                       <1>     pop     ds  ; **
1567 00001B85 07                       <1>     pop     es  ; * ; RECOVER SEGMENTS
1568 00001B86 CF                       <1>     iretd           ; ALL DONE
1569                                    <1>
1570                                    <1> set_txt_mode:
1571                                    <1>
1572                                    <1>     ; 29/07/2016
1573                                    <1>     ; 27/06/2016
1574 00001B87 B003                     <1>     mov     al, 3 ; 26/11/2020 (bit 7 = 0)
1575                                    <1>
1576                                    <1>     ; 17/11/2020 (TRDOS v2.0.3)
1577                                    <1>     ;mov    byte [noclearmem], 0
1578                                    <1>
1579                                    <1> ; 04/08/2022
1580                                    <1> ; 02/08/2022 (TRDOS 386 Kernel v2.0.5)
1581                                    <1> ; 12/04/2021
1582                                    <1> ; 10/08/2016
1583                                    <1> ; 08/08/2016
1584                                    <1> ; 30/07/2016
1585                                    <1> ; 29/07/2016
1586                                    <1> ; 25/07/2016 - 26/07/2016 - 27/07/2016
1587                                    <1> ; 07/07/2016 - 18/07/2016 - 23/07/2016
1588                                    <1> ; 02/07/2016 - 03/07/2016 - 04/07/2016
1589                                    <1> ; 26/06/2016
1590                                    <1> ; 24/06/2016 (set_txt_mode -> _set_mode)
1591                                    <1> ; 17/06/2016
1592                                    <1> ; 29/05/2016
1593                                    <1> ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
1594                                    <1>
1595                                    <1> _set_mode:
1596                                    <1>     ; 12/12/2020
1597                                    <1>     ; 26/11/2020
1598                                    <1>     ; call from 'biosfn_set_video_mode'
1599                                    <1>     ;       (bochs/plex86 video bios code)
1600                                    <1>     ; call from 'SET_MODE'
1601                                    <1>     ;       (TRDOS 386 v2 default, IBM PC/AT rom bios code)
1602                                    <1>     ; continue from 'set_txt_mode'
1603                                    <1>
1604                                    <1>     ; INPUT:
1605                                    <1>     ;       al = VGA video mode
1606                                    <1>     ; RETURN:
1607                                    <1>     ;       cf = 1 -> video mode not implemented
1608                                    <1>     ;       cf = 0 -> OK
1609                                    <1>     ;
1610                                    <1>     ; Modified registers: eax, bx, ecx, esi, edi, (ebp)
1611                                    <1>
1612                                    <1>     ; 17/11/2020 (TRDOS v2.0.3)
1613                                    <1>     ; no clear memory option
1614                                    <1>     ; (from mode number byte bit 7)
1615 00001B89 88C4                     <1>     mov     ah, al
1616 00001B8B 80E480                   <1>     and     ah, 80h
1617                                    <1>     ;mov    [noclearmem], al
1618 00001B8E 8825[17830100]           <1>     mov     [noclearmem], ah
1619                                    <1>     ;and    al, 7Fh ; clear bit 7
1620                                    <1>     ;;xor   [noclearmem], al ; clear bit 0 to 6
1621                                    <1>     ; 26/11/2020
1622 00001B94 30E0                     <1>     xor     al, ah ; and al, 7Fh
1623                                    <1>
1624                                    <1>     ; 19/11/2020
1625                                    <1>
1626                                    <1>     ; Video mode 03h action principle:
1627                                    <1>     ;
1628                                    <1>     ; for case 1:
1629                                    <1>     ; Current mode is 03h and next/requested mode is not 03h
1630                                    <1>     ;       - save mode (set mode 03h flag)
1631                                    <1>     ;       - save 8 video pages (which are will be restored)
1632                                    <1>     ;       - save active page number (which will be reactivated)
1633                                    <1>     ;       - set active page to 0 always (no multi screen)
1634                                    <1>     ;       - save 8 cursor positions (which will be restored)
1635                                    <1>     ;       - use 'noclearmem' option
1636                                    <1>     ;       [p_crt_mode] = 0 -> 03h
1637                                    <1>     ;
1638                                    <1>     ; for case 2:
1639                                    <1>     ; Current mode is 03h and next/requested mode is also 03h
1640                                    <1>     ;       - clear active video page if 'noclearmem' is not set
1641                                    <1>     ;       [p_crt_mode] = 0 -> 80h -> 0
1642                                    <1>     ;
1643                                    <1>     ; for case 3:
1644                                    <1>     ; Current mode is not 03h and next/requested mode is 03h
1645                                    <1>     ;       - restore video pages (8 video pages were saved)
1646                                    <1>     ;       - restore active page number (which were saved)
1647                                    <1>     ;       - restore 8 cursor positions (which were saved)
1648                                    <1>     ;       - reset/clear mode 03h flag
1649                                    <1>     ;       [p_crt_mode] = 03h -> 0
1650                                    <1>     ;
1651                                    <1>     ; for case 4:
1652                                    <1>     ; Current mode is not 03h and next/requested mode is not 03h
1653                                    <1>     ;       - use 'noclearmem' option
1654                                    <1>     ;       - set active page to 0 always
1655                                    <1>     ;       [p_crt_mode] = 03h -> 83h -> 03h
1656                                    <1>     ;
1657                                    <1>     ; initial (boot time) values:
1658                                    <1>     ;       [p_crt_mode] = 0 ("there isn't a page backup, yet")
1659                                    <1>     ;         [CRT_MODE] = 3 (kernel's starting mode)
1660                                    <1>
1661                                    <1>     ; 26/11/2020
1662 00001B96 3C03                     <1>     cmp     al, 03h     ; mode 3, 80x25 text, 16 colors
1663 00001B98 7515                     <1>     jne     short _sm_0 ; (default mode for TRDOS 386 mainprog)
```

```
1664                                    <1>
1665                                    <1>     ; case 2 or case 3
1666                                    <1>
1667                                    <1>     ; check current video mode if it is 03h
1668 00001B9A 08E4                     <1>     or      ah, ah ; 80h or 0 ('noclearmem' option)
1669 00001B9C 7521                     <1>     jnz     short _sm_1 ; do not clear display page
1670                                    <1>
1671                                    <1>     ; 26/11/2020
1672                                    <1>     ; Note:
1673                                    <1>     ; [CRT_MODE] = 0FFh for VESA VBE video modes
1674                                    <1>     ; [video_mode] = standard VGA and VESA VBE video modes
1675                                    <1>
1676 00001B9E 3805[1E680000]           <1>     cmp     [CRT_MODE], al ; 03h
1677 00001BA4 7520                     <1>     jne     short _sm_2    ; case 3 ([p_crt_mode] = 03h)
1678                                    <1>
1679                                    <1>     ; case 2
1680                                    <1>
1681                                    <1>     ; [p_crt_mode] = 0
1682                                    <1>
1683                                    <1>     ; 19/11/2020
1684                                    <1>     ; If '_set_mode' procedure is called for video mode 3
1685                                    <1>     ;     while video mode is 3, video page will be cleared
1686                                    <1>     ;     and cursor position of video page will be reset.
1687                                    <1>
1688                                    <1>     ; clear display page
1689 00001BA6 C605[15830100]80           <1>     mov     byte [p_crt_mode], 80h ; clear page sign
1690 00001BAD EB1C                     <1>     jmp     short _sm_3    ; bypass save video page routine
1691                                    <1> _sm_0:
1692                                    <1>     ; case 1 or case 4
1693                                    <1>
1694                                    <1>     ; 05/12/2020
1695 00001BAF 803D[1E680000]03         <1>     cmp     byte [CRT_MODE], 3 ; is current mode 03h?
1696 00001BB6 7507                     <1>     jne     short _sm_1    ; case 4 ; [p_crt_mode] = 03h
1697                                    <1>
1698                                    <1>     ; case 1
1699                                    <1>     ; [p_crt_mode] = 0
1700                                    <1>
1701                                    <1>     ; 19/11/2020
1702                                    <1>     ; If '_set_mode' procedure is called for a video mode
1703                                    <1>     ;     except video mode 3 while current video mode
1704                                    <1>     ;     is 3, all video pages of mode 3 will be copied
1705                                    <1>     ;     to 98000h address as backup, before mode change.
1706                                    <1>
1707                                    <1> _sm_save_pm:
1708                                    <1>     ; 12/12/2020
1709                                    <1>     ;; 03/07/2016
1710                                    <1>     ;; save video pages
1711                                    <1>     ;mov    esi, 0B8000h
1712                                    <1>     ;mov    edi, 98000h ; 30/07/2016
1713                                    <1>     ;mov    ecx, (0B8000h-0B0000h)/4
1714                                    <1>     ;rep    movsd
1715                                    <1>
1716                                    <1>     ;mov    byte [p_crt_mode], 3 ; previous mode, backup sign
1717                                    <1>     ;; 26/11/2020
1718                                    <1>     ;xchg   cl, [ACTIVE_PAGE]
1719                                    <1>     ;mov    [p_crt_page], cl  ; save as previous active page
1720                                    <1>     ;
1721                                    <1>     ;; save cursor positions
1722                                    <1>     ;mov    esi, CURSOR_POSN
1723                                    <1>     ;mov    edi, cursor_pposn ; cursor positions backup
1724                                    <1>     ;mov    cl, 4
1725                                    <1>     ;rep    movsd
1726                                    <1>
1727                                    <1>     ; 12/12/2020
1728 00001BB8 E86E020000               <1>     call    save_mode3_multiscreen
1729                                    <1>
1730                                    <1>     ; 29/07/2016
1731                                    <1>     ;;mov   [ACTIVE_PAGE], cl ; 0
1732                                    <1>     ;xchg   cl, [ACTIVE_PAGE]
1733                                    <1>     ;mov    [p_crt_page], cl  ; previous page (for mode 3)
1734                                    <1>
1735                                    <1>     ; [ACTIVE_PAGE] = 0
1736                                    <1>
1737 00001BBD EB07                     <1>     jmp     short _sm_2    ; case 1 - 19/11/2020
1738                                    <1> _sm_1:
1739                                    <1>     ; 26/11/2020
1740                                    <1>     ; 19/11/2020
1741                                    <1>
1742                                    <1>     ; case 4
1743 00001BBF 800D[15830100]80         <1>     or      byte [p_crt_mode], 80h
1744                                    <1>                             ; here [p_crt_mode] must be 83h
1745                                    <1>                             ;       (for case 4)
1746                                    <1>                             ; (because video mode 03h
1747                                    <1>                             ; was changed before as in case 1)
1748                                    <1>
1749                                    <1> _sm_2:      ; case 4 (jump to _sm_2) - 19/11/2020
1750                                    <1>
1751                                    <1>     ; 19/11/2020
1752                                    <1>     ; case 3
1753                                    <1>     ; If '_set_mode' procedure is called for video mode 3
1754                                    <1>     ;     while video mode is not 3 and if there is video
1755                                    <1>     ;     page backup for video mode 3, all (of 8) mode 3
1756                                    <1>     ;     video pages will be restored from 98000h.
1757                                    <1>
1758 00001BC6 A2[1E680000]              <1>     mov     [CRT_MODE], al  ; save mode in global variable
1759                                    <1> _sm_3:
1760                                    <1>     ; 04/08/2022 (TRDOS 386 v2.0.5)
1761                                    <1>     ; 30/07/2016
1762                                    <1>     ; 26/07/2016
1763                                    <1>     ; 25/07/2016
1764                                    <1>     ; set_mode_vga:
1765                                    <1>     ; 18/07/2016
1766                                    <1>     ; 14/07/2016
1767                                    <1>     ; 09/07/2016
1768                                    <1>     ; 04/07/2016
1769                                    <1>     ; 03/07/2016 (TRDOS 386 = TRDOS v2.0)
1770                                    <1>     ; /// video mode 13h ///
1771                                    <1>     ; derived from 'Plex86/Bochs VGABios' source code
1772                                    <1>     ; vgabios-0.7a (2011)
1773                                    <1>     ; by the LGPL VGABios developers Team (2001-2008)
1774                                    <1>     ; 'vgabios.c', 'vgatables.h'
1775                                    <1>     ;
1776                                    <1>     ; Oracle VirtualBox 5.0.24 VGABios Source Code
1777                                    <1>     ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
1778                                    <1>     ;
1779 00001BCB 88C4                     <1>     mov     ah, al
1780 00001BCD B910000000               <1>     mov     ecx, vga_mode_count
1781 00001BD2 BE[3A680000]             <1>     mov     esi, vga_modes
1782 00001BD7 31DB                     <1>     xor     ebx, ebx
1783                                    <1> _sm_4:
1784 00001BD9 AC                       <1>     lodsb
1785 00001BDA 38C4                     <1>     cmp     ah, al
1786 00001BDC 7406                     <1>     je      short _sm_5
1787 00001BDE FEC3                     <1>     inc     bl
```

```
1788 00001BE0 E2F7                <1>     loop    _sm_4
1789                              <1>
1790                              <1>     ; UNIMPLEMENTED VIDEO MODE !
1791                              <1>     ;xor    eax, eax
1792                              <1>         ;mov       [video_eax], eax ; 0
1793                              <1>
1794                              <1>     ; 26/11/2020
1795 00001BE2 F9                  <1>     stc     ; unimplemented video mode ! (cf=1)
1796                              <1>
1797 00001BE3 C3                  <1>     retn
1798                              <1>
1799                              <1> ;-----      EBX POINTS TO CORRECT ROW OF INITIALIZATION TABLE
1800                              <1>
1801                              <1> _sm_5:      ; 25/07/2016
1802                              <1>     ;mov    esi, ebx
1803                              <1>     ;add    esi, vga_memmodel
1804                              <1>     ;mov    al, [esi]
1805                              <1>     ; 19/11/2020
1806 00001BE4 8A83[8A680000]      <1>     mov     al, [ebx+vga_memmodel]
1807 00001BEA A2[2E830100]        <1>     mov     [VGA_MTYPE], al
1808                              <1>
1809 00001BEF 89DF                <1>     mov     edi, ebx
1810 00001BF1 81C7[9A680000]      <1>     add     edi, vga_dac_s
1811 00001BF7 C0E302              <1>     shl     bl, 2 ; byte -> dword
1812 00001BFA 81C3[4A680000]      <1>     add     ebx, vga_mode_tbl_ptr
1813                              <1>
1814                              <1>     ;mov    dword [VGA_BASE], 0B8000h
1815                              <1>     ;cmp    ah, 0Dh ; [CRT_MODE]
1816                              <1>     ;jb     short M9
1817                              <1>     ;mov    dword [VGA_BASE], 0A0000h
1818                              <1> ;M9:
1819 00001C00 8B33                <1>     mov     esi, [ebx]
1820 00001C02 89F3                <1>     mov     ebx, esi
1821 00001C04 83C614              <1>     add     esi, vga_p_cm_pos ; ebx + 20
1822 00001C07 668B06              <1>     mov     ax, [esi]       ; get the cursor mode from the table
1823 00001C0A 66A3[37680000]      <1>     mov     [CURSOR_MODE], ax ; save cursor mode (initial value)
1824                              <1>     ; al = 6, ah = 7
1825                              <1>     ; al = 0Dh, ah = 0Eh ; 25/07/2016
1826 00001C10 E893020000          <1>     call    cursor_shape_fix
1827                              <1>     ; al = 14, ah = 15 (If [CHAR_HEIGHT] = 16)
1828 00001C15 668906              <1>     mov     [esi], ax
1829                              <1>
1830 00001C18 56                  <1>     push    esi ; *
1831                              <1>
1832                              <1>     ; 17/04/2021
1833 00001C19 B603                <1>     mov     dh, 03h
1834                              <1>     ;
1835 00001C1B 8A25[25680000]      <1>     mov     ah, [VGA_MODESET_CTL]
1836 00001C21 80E408              <1>     and     ah, 8 ; default palette loading ?
1837 00001C24 7520                <1>     jnz     short _sm_6
1838                              <1>     ;mov    dx, 3C6h ; VGAREG_PEL_MASK (DAC mask register)
1839                              <1>     ; 17/04/2021
1840 00001C26 B2C6                <1>     mov     dl, 0C6h
1841 00001C28 B0FF                <1>     mov     al, 0FFh ; PEL mask
1842 00001C2A EE                  <1>     out     dx, al
1843 00001C2B 8A27                <1>     mov     ah, [edi] ; DAC model (selection number)
1844 00001C2D E8F80F0000          <1>     call    load_dac_palette
1845                              <1>     ; ecx = 0
1846 00001C32 F605[25680000]02    <1>     test    byte [VGA_MODESET_CTL], 2 ; gray scale summing
1847 00001C39 740B                <1>     jz      short _sm_6
1848 00001C3B 53                  <1>     push    ebx
1849 00001C3C 29DB                <1>     sub     ebx, ebx ; sub bl, bl
1850                              <1>     ;mov    cx, 256
1851                              <1>     ; 02/08/2022
1852                              <1>     ;sub    ecx, ecx
1853                              <1>     ; ecx = 0
1854 00001C3E FEC5                <1>     inc     ch
1855                              <1>     ; ecx = 256
1856 00001C40 E837100000          <1>     call    gray_scale_summing
1857 00001C45 5B                  <1>     pop     ebx
1858                              <1> _sm_6:
1859                              <1>     ; Reset Attribute Ctl flip-flop
1860                              <1>     ;mov    dx, 3DAh ; VGAREG_ACTL_RESET
1861                              <1>     ; 17/03/2021
1862 00001C46 B2DA                <1>     mov     dl, 0DAh ; dx = 3DAh
1863 00001C48 EC                  <1>     in      al, dx
1864                              <1>     ; Set Attribute Ctl
1865 00001C49 89DE                <1>     mov     esi, ebx ; addr of params tbl for selected mode
1866 00001C4B 83C623              <1>     add     esi, 35  ; actl regs
1867 00001C4E 30E4                <1>     xor     ah, ah ; 0
1868                              <1>     ;mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
1869                              <1>     ; 17/04/2021
1870 00001C50 B2C0                <1>     mov     dl, 0C0h
1871                              <1> _sm_7:
1872 00001C52 88E0                <1>     mov     al, ah
1873 00001C54 EE                  <1>     out     dx, al ; index
1874 00001C55 AC                  <1>     lodsb
1875                              <1>     ; DX = 3C0h = VGAREG_ACTL_WRITE_DATA
1876 00001C56 EE                  <1>     out     dx, al ; value
1877 00001C57 FEC4                <1>     inc     ah
1878 00001C59 80FC14              <1>     cmp     ah, 20 ; number of actl registers
1879 00001C5C 72F4                <1>     jb      short _sm_7
1880                              <1>     ;
1881 00001C5E 88E0                <1>     mov     al, ah ; 20
1882 00001C60 EE                  <1>     out     dx, al ; index
1883 00001C61 28C0                <1>     sub     al, al ; 0
1884 00001C63 EE                  <1>     out     dx, al ; value
1885                              <1>     ;
1886                              <1>     ; Set Sequencer Ctl
1887 00001C64 89DE                <1>     mov     esi, ebx ; addr of params tbl for selected mode
1888 00001C66 83C605              <1>     add     esi, 5 ; sequ regs
1889                              <1>     ;
1890                              <1>     ;mov    dx, 3C4h  ; VGAREG_SEQU_ADDRESS
1891                              <1>     ; 17/04/2021
1892 00001C69 B2C4                <1>     mov     dl, 0C4h
1893 00001C6B EE                  <1>     out     dx, al ; 0
1894                              <1>     ;inc    dx ; 3C5h ; VGAREG_SEQU_DATA
1895                              <1>     ; 17/04/2021
1896 00001C6C FEC2                <1>     inc     dl ; dx = 3C5h
1897 00001C6E B003                <1>     mov     al, 3
1898 00001C70 EE                  <1>     out     dx, al
1899 00001C71 B401                <1>     mov     ah, 1
1900                              <1> _sm_8:
1901 00001C73 88E0                <1>     mov     al, ah
1902                              <1>     ;mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
1903                              <1>     ;dec    dx
1904                              <1>     ; 17/04/2021
1905 00001C75 FECA                <1>     dec     dl ; dx = 3C4h
1906 00001C77 EE                  <1>     out     dx, al ; index
1907 00001C78 AC                  <1>     lodsb
1908                              <1>     ;inc    dx ; 3C5h ; VGAREG_SEQU_DATA
1909                              <1>     ; 17/04/2021
1910 00001C79 FEC2                <1>     inc     dl
1911 00001C7B EE                  <1>     out     dx, al
```

```
1912 00001C7C 80FC04              <1>     cmp     ah, 4 ; number of sequ regs
1913 00001C7F 7304                <1>     jnb     short _sm_9
1914 00001C81 FEC4                <1>     inc     ah
1915 00001C83 EBEE                <1>     jmp     short _sm_8
1916                              <1> _sm_9:
1917                              <1>     ; Set Grafx Ctl
1918 00001C85 89DE                <1>     mov     esi, ebx ; addr of params tbl for selected mode
1919 00001C87 83C637              <1>     add     esi, 55 ; grdc regs
1920 00001C8A 30E4                <1>     xor     ah, ah ; 0
1921                              <1> _sm_10:
1922 00001C8C 88E0                <1>     mov     al, ah
1923                              <1>     ;mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
1924                              <1>     ; 17/04/2021
1925 00001C8E B2CE                <1>     mov     dl, 0CEh
1926 00001C90 EE                  <1>     out     dx, al
1927 00001C91 AC                  <1>     lodsb
1928                              <1>     ;inc    dx ; 3CFh ; VGAREG_GRDC_DATA
1929                              <1>     ; 17/04/2021
1930 00001C92 FEC2                <1>     inc     dl ; 3CFh
1931 00001C94 EE                  <1>     out     dx, al
1932 00001C95 FEC4                <1>     inc     ah
1933 00001C97 80FC09              <1>     cmp     ah, 9 ; number of grdc regs
1934 00001C9A 72F0                <1>     jb      short _sm_10
1935                              <1>     ;
1936                              <1>     ; Disable CRTC write protection
1937                              <1>     ;mov    dx, 3D4h  ; VGAREG_VGA_CRTC_ADDRESS
1938                              <1>     ; 17/04/2021
1939 00001C9C B2D4                <1>     mov     dl, 0D4h
1940                              <1>     ;mov    al, 11h
1941                              <1>     ;out    dx, al
1942                              <1>     ;inc    dx
1943                              <1>     ;sub    al, al
1944                              <1>     ;out    dx, al
1945 00001C9E 66B81100            <1>     mov     ax, 11h
1946 00001CA2 66EF                <1>     out     dx, ax
1947 00001CA4 89DE                <1>     mov     esi, ebx ; addr of params tbl for selected mode
1948 00001CA6 83C60A              <1>     add     esi, 10 ; crtc regs
1949                              <1>     ; ah = 0
1950                              <1> _sm_11:
1951 00001CA9 88E0                <1>     mov     al, ah
1952                              <1>     ; dx = 3D4h = VGAREG_VGA_CRTC_ADDRESS
1953 00001CAB EE                  <1>     out     dx, al ; index
1954 00001CAC AC                  <1>     lodsb
1955                              <1>     ;inc    dx  ; VGAREG_VGA_CRTC_ADDRESS + 1
1956                              <1>     ; 17/04/2021
1957 00001CAD FEC2                <1>     inc     dl
1958 00001CAF EE                  <1>     out     dx, al ; value
1959 00001CB0 80FC18              <1>     cmp     ah, 24 ; number of crtc registers - 1
1960 00001CB3 7306                <1>     jnb     short _sm_12
1961 00001CB5 FEC4                <1>     inc     ah
1962                              <1>     ;dec    dx ; 3D4h
1963                              <1>     ; 17/04/2021
1964 00001CB7 FECA                <1>     dec     dl
1965 00001CB9 EBEE                <1>     jmp     short _sm_11
1966                              <1> _sm_12:
1967                              <1>     ; Set the misc register
1968                              <1>     ;mov    dx, 3CCh ; VGAREG_READ_MISC_OUTPUT
1969                              <1>     ; 17/04/2021
1970 00001CBB B2CC                <1>     mov     dl, 0CCh
1971 00001CBD 8A4309              <1>     mov     al, [ebx+9] ; misc reg
1972 00001CC0 EE                  <1>     out     dx, al
1973                              <1>     ;
1974                              <1>     ; Enable video
1975                              <1>     ;mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
1976                              <1>     ; 17/04/2021
1977 00001CC1 B2C0                <1>     mov     dl, 0C0h
1978 00001CC3 B020                <1>     mov     al, 20h
1979 00001CC5 EE                  <1>         out     dx, al   ; set bit 5 to 1
1980                              <1>     ;mov    dx, 3DAh ; VGAREG_ACTL_RESET
1981                              <1>     ; 17/04/2021
1982 00001CC6 B2DA                <1>     mov     dl, 0DAh
1983 00001CC8 EC                  <1>     in      al, dx
1984                              <1>     ;
1985                              <1>     ; 17/11/2020
1986                              <1>     ;cmp    byte [noclearmem], 0
1987                              <1>         ;ja short _sm_15
1988                              <1>
1989 00001CC9 F605[17830100]80    <1>     test    byte [noclearmem], 80h
1990 00001CD0 753B                <1>     jnz     short _sm_15
1991                              <1>
1992                              <1>     ; 29/07/2016
1993 00001CD2 31C0                <1>     xor     eax, eax
1994                              <1>     ;mov    ecx, 4000h ; 16K words (32K)
1995                              <1>     ; 02/08/2022
1996 00001CD4 29C9                <1>     sub     ecx, ecx
1997 00001CD6 B540                <1>     mov     ch, 40h
1998                              <1>     ; ecx = 4000h
1999 00001CD8 803D[2E830100]02    <1>     cmp     byte [VGA_MTYPE], 2  ; CTEXT, MTEXT, CGA
2000 00001CDF 7715                <1>     ja      short _sm_14    ; no ? (0A0000h)
2001 00001CE1 BF00800B00          <1>     mov     edi, 0B8000h
2002 00001CE6 7409                <1>     je      short _sm_13   ; CGA graphics mode
2003                              <1>     ; 08/08/2016
2004 00001CE8 A3[2A830100]        <1>     mov     [VGA_INT43H], eax ; 0 ; default font
2005 00001CED 66B82007            <1>     mov     ax, 0720h      ; CGA text mode
2006                              <1> _sm_13:
2007 00001CF1 F366AB              <1>     rep     stosw
2008 00001CF4 EB17                <1>     jmp     short _sm_15
2009                              <1>
2010                              <1> _sm_14:
2011 00001CF6 BF00000A00          <1>     mov     edi, 0A0000h
2012                              <1>     ; ecx = 16384 dwords (64K)
2013                              <1>     ;mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
2014                              <1>     ; 17/04/2021
2015 00001CFB B2C4                <1>     mov     dl, 0C4h
2016 00001CFD B002                <1>     mov     al, 2
2017 00001CFF EE                  <1>     out     dx, al
2018                              <1>     ;mov    dx, 3C5h ; VGAREG_SEQU_DATA
2019                              <1>     ;inc    dx
2020                              <1>     ; 17/04/2021
2021 00001D00 FEC2                <1>     inc     dl ; 3C5h
2022 00001D02 EC                  <1>     in      al, dx ; mmask
2023                              <1>     ;push   ax
2024                              <1>     ; 12/04/2021
2025 00001D03 50                  <1>     push    eax
2026 00001D04 B00F                <1>     mov     al, 0Fh ; all planes
2027 00001D06 EE                  <1>     out     dx, al
2028 00001D07 30C0                <1>     xor     al, al ; 0
2029 00001D09 F3AB                <1>     rep     stosd  ; ecx = 163684 (64K)
2030                              <1>     ;pop    ax
2031                              <1>     ; 12/04/2021
2032 00001D0B 58                  <1>     pop     eax
2033 00001D0C EE                  <1>     out     dx, al  ; mmask
2034                              <1> _sm_15:
2035                              <1>     ; ebx = addr of params tbl for selected mode
```

```
                                <1>     ; 10/08/2016
00001D0D 668B03                 <1>     mov     ax, [ebx] ; num of columns, 'twidth'
00001D10 A2[20680000]           <1>     mov     [CRT_COLS], al
                                <1>     ;; 26/07/2016
                                <1>     ;; CRTC_ADDRESS = 3D4h (always)
                                <1>     ;mov    ah, [ebx+1] ; num of rows, 'theightm1'
00001D15 FEC4                   <1>     inc     ah ; 09/07/2016
00001D17 8825[26680000]         <1>     mov     [VGA_ROWS], ah
                                <1>     ; 10/08/2016
00001D1D 8A4302                 <1>     mov     al, [ebx+2]
00001D20 A2[22680000]           <1>     mov     [CHAR_HEIGHT], al
                                <1>     ; 29/07/2016
                                <1>     ; length of regen buffer in bytes
00001D25 668B4B03               <1>     mov     cx, [ebx+3] ; 'slength_l'
00001D29 66890D[18830100]       <1>     mov     [CRT_LEN], cx
                                <1>     ;
                                <1>     ; 27/07/2016
00001D30 30E4                   <1>     xor     ah, ah
00001D32 A0[AE760100]           <1>     mov     al, [ACTIVE_PAGE] ; may be > 0 for mode 3
                                <1>     ;mul    word [CRT_LEN] ; 4096 for mode 3
00001D37 66F7E1                 <1>     mul     cx ; 29/07/2016
00001D3A 66A3[9C760100]         <1>     mov     [CRT_START], ax
                                <1>     ;
00001D40 B060                   <1>     mov     al, 60h
                                <1>     ;cmp    byte [noclearmem], 0
                                <1>     ;jna    short _sm_16
                                <1>     ;add    al, 80h
00001D42 0A05[17830100]         <1>     or      al, [noclearmem] ; 17/11/2020
                                <1> _sm_16:
00001D48 A2[23680000]           <1>     mov     [VGA_VIDEO_CTL], al
00001D4D C605[24680000]F9       <1>     mov     byte [VGA_SWITCHES], 0F9h
00001D54 8025[25680000]7F       <1>     and     byte [VGA_MODESET_CTL], 7Fh
                                <1>
00001D5B 5E                     <1>     pop     esi ; *
                                <1>
                                <1>     ; 26/07/2016
                                <1>     ; 07/07/2016
00001D5C 668B0D[37680000]       <1>     mov     cx, [CURSOR_MODE] ; restore cursor mode (initial value)
00001D63 66870E                 <1>     xchg    cx, [esi] ; cl = start line, ch = end line
                                <1>                       ; reset to initial value
00001D66 86CD                   <1>     xchg    ch, cl  ; ch = start line, cl = end line
00001D68 66890D[37680000]       <1>     mov     [CURSOR_MODE], cx ; save (fixed) cursor mode
                                <1>
                                <1>     ; 27/07/2016
00001D6F 803D[2E830100]02       <1>     cmp     byte [VGA_MTYPE], 2 ; CTEXT, MTEXT
00001D76 7317                   <1>     jnb     short _sm_17
                                <1>
                                <1>     ; Set cursor shape
                                <1>     ;mov    cx, 0607h
                                <1>     ;call   _set_ctype
                                <1>
                                <1>     ; 29/07/2016
00001D78 B40A                   <1>     mov     ah, 10  ; 6845 register for cursor set
00001D7A E81F060000             <1>     call    m16     ; output cx register
                                <1>
                                <1>     ; 25/07/2016
00001D7F 803D[1E680000]03       <1>         cmp     byte [CRT_MODE], 03h
00001D86 7507                   <1>     jne     short _sm_17
                                <1>     ; 26/07/2016
                                <1>
00001D88 A0[AE760100]           <1>     mov     al, [ACTIVE_PAGE]
00001D8D EB0B                   <1>     jmp     short _sm_18
                                <1> _sm_17:
                                <1>     ; Set cursor pos for page 0..7
                                <1>     ;sub    ax, ax ; eax = 0
00001D8F 29C0                   <1>     sub     eax, eax ; 17/11/2020
00001D91 BF[9E760100]           <1>     mov     edi, CURSOR_POSN
00001D96 AB                     <1>     stosd
00001D97 AB                     <1>     stosd
00001D98 AB                     <1>     stosd
00001D99 AB                     <1>     stosd
                                <1>     ;; Set active page 0
                                <1>     ;mov    [ACTIVE_PAGE], al ; 0
                                <1> _sm_18:
                                <1>     ; 29/07/2016
00001D9A 803D[2E830100]02       <1>     cmp     byte [VGA_MTYPE], 2 ; CTEXT, MTEXT
                                <1>     ;jnb    _sm_23
                                <1>     ; 04/08/2022
00001DA1 7205                   <1>     jb      short _sm_24
00001DA3 E90C020000             <1>     jmp     _set_active_page
                                <1> _sm_24:
                                <1>     ;cmp    byte [CHAR_HEIGHT], 16
                                <1>     ;je     short _sm_19
                                <1>
                                <1>     ;; copy and activate 8x16 font
                                <1>
                                <1>     ; 26/07/2016
00001DA8 B004                   <1>     mov     al, 04h
                                <1>     ;sub    bl, bl
                                <1>     ; AX = 1104H ; Load ROM 8x16 Character Set
                                <1>     ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
00001DAA E844160000             <1>     call    load_text_8_16_pat
                                <1>
                                <1>     ; video_func_1103h:
                                <1>     ; biosfn_set_text_block_specifier:
                                <1>     ; BL = font block selector code
                                <1>     ; NOTE: TRDOS 386 only uses and sets font block 0
                                <1>     ; (It is as BL = 0 for TRDOS 386)
00001DAF 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
00001DB3 66B80300               <1>     mov     ax, 03h
00001DB7 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>     ;
00001DB9 30C0                   <1>     xor     al, al
00001DBB 3805[15830100]         <1>         cmp     [p_crt_mode], al ; 0
00001DC1 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>
00001DC3 E9EC010000         <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
00001DC8 803D[1E680000]03   <1>     cmp    byte [CRT_MODE], 3  ; new video mode
00001DCF 754E               <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
00001DD1 A0[16830100]       <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
00001DD6 F605[15830100]7F   <1>     test    byte [p_crt_mode], 7Fh ; 83h or 80h or 03h
00001DDD 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
00001DDF A2[AE760100]       <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
00001DE4 E89A000000         <1>     call   _restore_mode3_multiscreen
                            <1>
                            <1>     ;jmp   short _sm_23 ; do not clear current video pages
                            <1>
                            <1>     ; 19/11/2020
00001DE9 E9C6010000         <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
00001DEE F605[17830100]80   <1>     test   byte [noclearmem], 80h
00001DF5 7528               <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
00001DF7 66B82007           <1>     mov    ax, 0720h
                            <1>     ; 26/11/2020
                            <1>     ;mov   ecx, 2048 ; 4096/2
                            <1>     ; 02/08/2022
00001DFB 29C9               <1>     sub    ecx, ecx
00001DFD B508               <1>     mov    ch, 08h
                            <1>     ; ecx = 0800h
00001DFF BF00800B00         <1>     mov    edi, 0B8000h ; [crt_base]
00001E04 66033D[9C760100]   <1>     add    di, [CRT_START]
00001E0B F366AB             <1>     rep    stosw  ; FILL THE REGEN BUFFER WITH BLANKS
                            <1>     ;
                            <1>     ; 19/11/2020
00001E0E A0[AE760100]       <1>     mov    al, [ACTIVE_PAGE] ; 0 to 7 (for video mode 3)
00001E13 0FB6D8             <1>     movzx  ebx, al
00001E16 D0E3               <1>     shl    bl, 1
00001E18 66898B[9E760100]   <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
00001E1F 8025[15830100]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
00001E26 E989010000             <1>     jmp     _set_active_page
                                <1>
                                <1> ;-----      NORMAL RETURN FROM ALL VIDEO RETURNS
                                <1>     ;retn
                                <1>
                                <1> save_mode3_multiscreen:
                                <1>     ; 02/08/2022 (TRDOS v2.0.5)
                                <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
00001E2B BE00800B00             <1>     mov     esi, 0B8000h
00001E30 BF00800900             <1>     mov     edi, 98000h ; 30/07/2016
                                <1>     ;mov    ecx, (0B8000h-0B0000h)/4
                                <1>     ; 02/08/2022
00001E35 29C9                   <1>     sub     ecx, ecx
00001E37 B520                   <1>     mov     ch, 20h
                                <1>     ; ecx = 2000h
00001E39 F3A5                   <1>     rep     movsd
                                <1>
00001E3B C605[15830100]03       <1>     mov     byte [p_crt_mode], 3 ; previous mode, backup sign
                                <1>     ; 26/11/2020
00001E42 860D[AE760100]         <1>     xchg    cl, [ACTIVE_PAGE]
00001E48 880D[16830100]         <1>     mov     [p_crt_page], cl  ; save as previous active page
                                <1>
                                <1>     ; save cursor positions
00001E4E BE[9E760100]           <1>     mov     esi, CURSOR_POSN
00001E53 BF[1A830100]           <1>     mov     edi, cursor_pposn ; cursor positions backup
00001E58 B104                   <1>     mov     cl, 4
00001E5A F3A5                   <1>     rep     movsd
00001E5C C3                     <1>     retn
                                <1>
                                <1> restore_mode3_multiscreen:
                                <1>     ; 02/08/2022 (TRDOS v2.0.5)
                                <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>
00001E5D A0[16830100]           <1>     mov     al, [p_crt_page] ; previous mode 3 active page
00001E62 A2[AE760100]           <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
                                <1>     ; restore CRT_START according to ACTIVE_PAGE
                                <1>     ;mov    [CRT_START], cx ; 0
                                <1>     ; 12/12/2020
00001E67 66C705[9C760100]00-    <1>     mov     word [CRT_START], 0
00001E6F 00                     <1>
                                <1>
                                <1>     ; check active page and set it again if it is not 0
00001E70 08C0                   <1>     or      al, al
                                <1>     ;;jz    short vbe3_pmi_7
                                <1>     ;jz     short _restore_mode3_multiscreen
00001E72 740F                   <1>     jz      short r_m3_ms_1
00001E74 88C1                   <1>     mov     cl, al
                                <1> ;vbe3_pmi_5:
                                <1> r_m3_ms_0:
00001E76 668105[9C760100]00-    <1>     add     word [CRT_START], 4096
00001E7E 10                     <1>
00001E7F FEC9                   <1>     dec     cl
                                <1>     ;jnz    short vbe3_pmi_5
00001E81 75F3                   <1>     jnz     short r_m3_ms_0
                                <1> r_m3_ms_1:
                                <1>     ; 12/12/2020
                                <1>     ; moved here from '_set_mode'
                                <1> _restore_mode3_multiscreen:
                                <1>     ; Modified registers: ecx, esi, edi
                                <1>
                                <1>     ; restore video pages
00001E83 BE00800900             <1>     mov     esi, 98000h ; 30/07/2016
00001E88 BF00800B00             <1>     mov     edi, 0B8000h
                                <1>     ;mov    cx, 2000h ; 8K dwords (32K)
                                <1>     ;mov    ecx, 2000h
                                <1>     ; 02/08/2022
00001E8D 29C9                   <1>     sub     ecx, ecx
00001E8F B520                   <1>     mov     ch, 20h
                                <1>     ; ecx = 2000h
00001E91 F3A5                   <1>     rep     movsd
                                <1>
                                <1>     ; 19/11/2020
00001E93 880D[15830100]         <1>     mov     [p_crt_mode], cl ; reset ('case 3' end condition)
                                <1>
                                <1>     ; restore cursor positions
00001E99 BE[1A830100]           <1>     mov     esi, cursor_pposn
00001E9E BF[9E760100]           <1>     mov     edi, CURSOR_POSN
                                <1>     ;mov    ecx, 4  ; restore all cursor positions (16 bytes)
00001EA3 B104                   <1>     mov     cl, 4
00001EA5 F3A5                   <1>     rep     movsd
00001EA7 C3                     <1>     retn
                                <1>
                                <1> cursor_shape_fix:
                                <1>     ; 12/04/2021
                                <1>     ; 07/07/2016
                                <1>     ; (Cursor start and cursor end line values -6,7-
                                <1>     ; will be fixed depending on character height)
                                <1>     ;
                                <1>     ; derived from 'Plex86/Bochs VGABios' source code
                                <1>     ; vgabios-0.7a (2011)
                                <1>     ; by the LGPL VGABios developers Team (2001-2008)
                                <1>     ; 'vgabios.c', ' biosfn_set_cursor_shape (CH,CL)'
```

```
                                <1>     ;
                                <1>     ; INPUT ->
                                <1>     ;       AL = cursor start line (=6)
                                <1>     ;       AH = cursor end line (=7)
                                <1>     ; OUTPUT ->
                                <1>     ;       AL = cursor start line (=14)
                                <1>     ;       AH = cursor end line (=15)
                                <1>     ;
                                <1>     ;; if((modeset_ctl&0x01)&&(cheight>8)&&(CL<8)&&(CH<0x20))
                                <1>     ;
                                <1>     ;test  byte [VGA_MODESET_CTL], 1 ; VGA active
                                <1>     ;jz    short csf_3
00001EA8 803D[22680000]08       <1>     cmp    byte [CHAR_HEIGHT], 8
00001EAF 7647                   <1>     jna    short csf_3
00001EB1 80FC08                 <1>     cmp    ah, 8
00001EB4 7342                   <1>     jnb    short csf_3
00001EB6 3C20                   <1>     cmp    al, 20h
00001EB8 733E                   <1>     jnb    short csf_3
                                <1>     ;
                                <1>     ;push  ax
                                <1>     ; 12/04/2021
00001EBA 50                     <1>     push   eax
                                <1>     ; {
                                <1>     ; if(CL!=(CH+1))
00001EBB FEC0                   <1>     inc    al
00001EBD 38C4                   <1>     cmp    ah, al   ; ah != al + 1
00001EBF 740F                   <1>         je     short csf_1
                                <1>     ; CH = ((CH+1) * cheight / 8) -1;
00001EC1 8A25[22680000]         <1>     mov    ah, [CHAR_HEIGHT]
00001EC7 F6E4                   <1>     mul    ah
00001EC9 C0E803                 <1>     shr    al, 3 ; / 8
00001ECC FEC8                   <1>     dec    al ; - 1
00001ECE EB0E                   <1>     jmp    short csf_2
                                <1> csf_1:
                                <1>     ; }
                                <1>     ; else         ; ah = al + 1
                                <1>     ; {
00001ED0 FEC4                   <1>     inc    ah      ; ah = ah + 1
                                <1>     ; CH = ((CL+1) * cheight / 8) - 2;
00001ED2 A0[22680000]           <1>     mov    al, [CHAR_HEIGHT]
00001ED7 F6E4                   <1>     mul    ah
00001ED9 C0E803                 <1>     shr    al, 3 ; / 8
00001EDC 2C02                   <1>     sub    al, 2 ; - 2
                                <1>     ; al = 14 (if [CHAR_HEIGHT] = 16)
                                <1> csf_2:
00001EDE 880424                 <1>     mov    [esp], al
00001EE1 8A642401               <1>     mov    ah, [esp+1]
                                <1>     ; CL = ((CL+1) * cheight / 8) - 1;
00001EE5 FEC4                   <1>     inc    ah
00001EE7 A0[22680000]           <1>     mov    al, [CHAR_HEIGHT]
00001EEC F6E4                   <1>     mul    ah
00001EEE C0E803                 <1>     shr    al, 3 ; / 8
00001EF1 FEC8                   <1>     dec    al ; - 1
00001EF3 88442401               <1>     mov    [esp+1], al
                                <1>     ; ah = 15 (if [CHAR_HEIGHT] = 16)
                                <1>     ;
                                <1>     ;pop   ax
                                <1>     ; 12/04/2021
00001EF7 58                     <1>     pop    eax
                                <1> csf_3:
00001EF8 C3                     <1>     retn
                                <1>
                                <1> SET_CTYPE:
                                <1>     ; 04/08/2022 (TRDOS 386 v2.0.5)
                                <1>     ; 12/09/2016
                                <1>     ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
00001EF9 803D[1E680000]07       <1>     cmp    byte [CRT_MODE], 7
                                <1>     ;ja    VIDEO_RETURN ; 12/09/2016
                                <1>     ; 04/08/2022
00001F00 7738                   <1>     ja     short set_cpos_inv_vp
00001F02 E805000000             <1>     call   _set_ctype
00001F07 E96CFCFFFF             <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>     ; 02/08/2022 (TRDOS 386 Kernel v2.0.5)
                                <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.)
                                <1>
                                <1>     ;mov   ax, cx
                                <1>     ; 02/08/2022
00001F0C 89C8                   <1>     mov    eax, ecx
                                <1>
00001F0E 86E0                   <1>     xchg   al, ah
                                <1>     ; AL = start line, AH = stop line
00001F10 E893FFFFFF             <1>     call   cursor_shape_fix
                                <1>     ; AL = start line (fixed), AH = stop line (fixed)
                                <1>     ;mov   cx, ax
                                <1>     ; 02/08/2022
00001F15 89C1                   <1>     mov    ecx, eax
00001F17 86CD                   <1>     xchg   ch, cl
                                <1>     ; CH = start line (fixed), CL = stop line (fixed)
                                <1>     ;
00001F19 B40A                   <1>     mov    ah, 10  ; 6845 register for cursor set
00001F1B 66890D[37680000]       <1>     mov    [CURSOR_MODE], cx ; save in data area
                                <1>     ;call  m16    ; output cx register
                                <1>     ;retn
00001F22 E977040000             <1>         jmp    m16
                                <1>
```

```
2530                                    <1> SET_CPOS:
2531                                    <1>     ; 02/08/2022 (TRDOS 386 Kernel v2.0.5)
2532                                    <1>     ; 12/09/2016
2533                                    <1>     ; 07/07/2016
2534                                    <1>     ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
2535 00001F27 80FF07                   <1>     cmp     bh, 7 ; video page > 7 ; 07/07/2016
2536                                    <1>     ;ja     VIDEO_RETURN
2537                                    <1>     ; 02/08/2022
2538 00001F2A 770E                     <1>     ja      short set_cpos_inv_vp
2539                                    <1>     ;
2540 00001F2C 803D[1E680000]07         <1>     cmp     byte [CRT_MODE], 7
2541 00001F33 770A                     <1>     ja      short vga_set_cpos ; 12/09/2016
2542 00001F35 E839040000               <1>     call    _set_cpos
2543                                    <1> set_cpos_inv_vp:    ; 02/08/2022
2544 00001F3A E939FCFFFF               <1>         jmp     VIDEO_RETURN
2545                                    <1>
2546                                    <1> vga_set_cpos:
2547                                    <1>     ; 12/09/2016
2548                                    <1>     ; 09/07/2016
2549                                    <1>     ; set cursor position
2550                                    <1>     ; NOTE: Hardware cursor position will not be set
2551                                    <1>     ;   in any VGA modes (>7)
2552                                    <1>     ;   But, cursor position will be saved into
2553                                    <1>     ;   [CURSOR_POSN].
2554                                    <1>     ;   TRDOS 386 (TRDOS v2.0) uses only one page
2555                                    <1>     ;   (page 0) for all graphics modes.
2556                                    <1>
2557 00001F3F 668915[9E760100]          <1>     mov     [CURSOR_POSN], dx ; save cursor pos for pg 0
2558                                    <1>     ; 04/08/2016
2559                                    <1>     ;mov    bh, [ACTIVE_PAGE] ; = 0
2560                                    <1>     ;call   _set_cpos
2561 00001F46 E92DFCFFFF               <1>     jmp     VIDEO_RETURN
2562                                    <1>
2563                                    <1> READ_CURSOR:
2564                                    <1>     ; 12/09/2016
2565                                    <1>     ; 07/07/2016
2566                                    <1>     ; 12/05/2016
2567                                    <1>     ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
2568                                    <1>     ;
2569                                    <1>     ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
2570                                    <1>
2571                                    <1> ;--------------------------------------------------
2572                                    <1> ; READ_CURSOR
2573                                    <1> ;   THIS ROUTINE READS THE CURRENT CURSOR VALUE FROM THE
2574                                    <1> ;   845, FORMATS IT, AND SENDS IT BACK TO THE CALLER
2575                                    <1> ; INPUT
2576                                    <1> ;   BH - PAGE OF CURSOR
2577                                    <1> ; OUTPUT
2578                                    <1> ;   DX - ROW, COLUMN OF THE CURRENT CURSOR POSITION
2579                                    <1> ;   CX - CURRENT CURSOR MODE
2580                                    <1> ;--------------------------------------------------
2581                                    <1>
2582                                    <1>     ; BH = Video page number (0 to 7)
2583                                    <1>
2584                                    <1>     ; 07/07/2016
2585 00001F4B 80FF07                   <1>     cmp     bh, 7 ; video page > 7 (invalid)
2586 00001F4E 7606                     <1>     jna     short read_cursor_1
2587                                    <1>     ; invalid video page (input)
2588 00001F50 31C9                     <1>     xor     ecx, ecx ; 0
2589 00001F52 31D2                     <1>     xor     edx, edx ; 0
2590 00001F54 EB15                     <1>     jmp     short read_cursor_2
2591                                    <1> read_cursor_1:
2592                                    <1>     ; 12/09/2016
2593 00001F56 803D[1E680000]07         <1>     cmp     byte [CRT_MODE], 7 ; vga mode
2594 00001F5D 7727                     <1>     ja      short vga_get_cpos
2595                                    <1>     ;
2596 00001F5F E815000000               <1>     call    get_cpos
2597 00001F64 0FB70D[37680000]         <1>     movzx   ecx, word [CURSOR_MODE]
2598                                    <1> read_cursor_2:
2599 00001F6B 5D                       <1>     pop     ebp
2600 00001F6C 5F                       <1>     pop     edi
2601 00001F6D 5E                       <1>     pop     esi
2602 00001F6E 5B                       <1>     pop     ebx
2603 00001F6F 58                       <1>     pop     eax  ; DISCARD SAVED CX AND DX
2604 00001F70 58                       <1>     pop     eax
2605 00001F71 A1[08830100]             <1>     mov     eax, [video_eax] ; 12/05/2016
2606                                    <1>     ;;15/01/2017
2607                                    <1>     ;;mov   byte [intflg], 0 ; 07/01/2017
2608 00001F76 1F                       <1>     pop     ds
2609 00001F77 07                       <1>     pop     es
2610 00001F78 CF                       <1>     iretd
2611                                    <1>
2612                                    <1> get_cpos:
2613                                    <1>     ; 12/05/2016
2614                                    <1>     ; 16/01/2016
2615                                    <1>     ; BH = Video page number (0 to 7)
2616                                    <1>     ;
2617 00001F79 D0E7                     <1>     shl     bh, 1 ; WORD OFFSET
2618 00001F7B 0FB6F7                   <1>     movzx   esi, bh
2619 00001F7E 0FB796[9E760100]         <1>     movzx   edx, word [esi+CURSOR_POSN]
2620 00001F85 C3                       <1>     retn
2621                                    <1>
2622                                    <1> vga_get_cpos:
2623                                    <1>     ; 12/09/2016
2624                                    <1>     ; get cursor position (vga)
2625 00001F86 0FB715[9E760100]         <1>     movzx   edx, word [CURSOR_POSN] ; cursor pos for pg 0
2626 00001F8D 31C9                     <1>     xor     ecx, ecx ; Cursor Mode = 0 (invalid)
2627 00001F8F EBDA                     <1>     jmp     short read_cursor_2
2628                                    <1>
2629                                    <1> ACT_DISP_PAGE:
2630                                    <1>     ; 07/07/2016
2631                                    <1>     ; 26/06/2016
2632                                    <1>     ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
2633                                    <1>     ;
2634                                    <1>     ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
2635                                    <1>     ;
2636                                    <1> ;--------------------------------------------------
2637                                    <1> ; ACT_DISP_PAGE
2638                                    <1> ;   THIS ROUTINE SETS THE ACTIVE DISPLAY PAGE, ALLOWING
2639                                    <1> ;   THE FULL USE OF THE MEMORY SET ASIDE FOR THE VIDEO ATTACHMENT
2640                                    <1> ; INPUT
2641                                    <1> ;   AL HAS THE NEW ACTIVE DISPLAY PAGE
2642                                    <1> ; OUTPUT
2643                                    <1> ;   THE 6845 IS RESET TO DISPLAY THAT PAGE
2644                                    <1> ;--------------------------------------------------
2645                                    <1>     ; 07/07/2016
2646 00001F91 3C07                     <1>     cmp     al, 7   ; > 7 = invalid video page number
2647                                    <1>     ;ja     VIDEO_RETURN
2648 00001F93 7715                     <1>         ja  short adp_2 ; 18/11/2020
2649                                    <1>     ;cmp    byte [CRT_MODE], 3
2650                                    <1>     ;je     short adp_1
2651                                    <1>     ; 18/11/2020
2652 00001F95 8A25[1E680000]           <1>     mov     ah, [CRT_MODE]
2653 00001F9B 80FC03                   <1>     cmp     ah, 3
```

```
2654 00001F9E 7605              <1>     jna     short adp_1 ; mode 01h, 00h (01h), 02h (03h), 03h
2655 00001FA0 80FC07            <1>     cmp     ah, 7       ; mode 07h (03h)
2656 00001FA3 7505              <1>     jne     short adp_2
2657                            <1>     ;and    al, al
2658                            <1>         ;jnz        VIDEO_RETURN
2659                            <1>     ;;sub   al, al ; 0 ; force to page 0
2660                            <1> adp_1:
2661 00001FA5 E805000000        <1>     call    set_active_page
2662                            <1> adp_2:
2663 00001FAA E9C9FBFFFF        <1>         jmp     VIDEO_RETURN
2664                            <1>
2665                            <1> set_active_page:   ; tty_sw
2666                            <1>     ; 02/08/2022 (TRDOS 386 Kernel v2.0.5)
2667                            <1>     ; 09/12/2017
2668                            <1>     ; 26/07/2016
2669                            <1>     ; 26/06/2016
2670                            <1>     ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
2671                            <1>     ; 30/06/2015
2672                            <1>     ; 04/03/2014  (act_disp_page --> tty_sw)
2673                            <1>     ; 10/12/2013
2674                            <1>     ; 04/12/2013
2675                            <1>     ;
2676 00001FAF A2[AE760100]      <1>     mov     [ACTIVE_PAGE], al ; save active page value ; [ptty]
2677                            <1> _set_active_page:
2678                            <1>     ; 27/06/2015
2679                            <1>     ;movzx  ebx, al
2680                            <1>     ; 02/08/2022
2681 00001FB4 0FB6C0            <1>     movzx   eax, al
2682 00001FB7 89C3              <1>     mov     ebx, eax
2683                            <1>     ;
2684                            <1>     ;cbw    ; 07/09/2014 (ah=0)
2685                            <1>     ; 02/08/2022
2686                            <1>     ;sub    ah, ah ; 09/12/2017
2687 00001FB9 66F725[18830100]  <1>     mul     word [CRT_LEN]  ; get saved length of regen buffer
2688                            <1>                             ; display page times regen length
2689                            <1>     ; 10/12/2013
2690 00001FC0 66A3[9C760100]    <1>     mov     [CRT_START], ax ; save start address for later
2691                            <1>     ;mov    cx, ax ; start address to cx
2692                            <1>     ; 02/08/2022
2693 00001FC6 89C1              <1>     mov     ecx, eax
2694                            <1> _M16:
2695                            <1>     ;;sar   cx, 1
2696                            <1>     ;shr    cx, 1   ; divide by 2 for 6845 handling
2697                            <1>     ; 02/08/2022
2698 00001FC8 D1E9              <1>     shr     ecx, 1
2699 00001FCA B40C              <1>     mov     ah, 12  ; 6845 register for start address
2700 00001FCC E8CD030000        <1>     call    m16
2701                            <1>     ;sal    bx, 1
2702                            <1>     ; 01/09/2014
2703 00001FD1 D0E3              <1>     shl     bl, 1   ; *2 for word offset
2704 00001FD3 81C3[9E760100]    <1>     add     ebx, CURSOR_POSN
2705 00001FD9 668B13            <1>     mov     dx, [ebx] ; get cursor for this page
2706                            <1>     ; 16/01/2016
2707                            <1>     ;call   m18
2708                            <1>     ;retn
2709 00001FDC E9A9030000        <1>     jmp     m18
2710                            <1>
2711                            <1> position:
2712                            <1>     ; 02/08/2022 - TRDOS 386 v2.0.5
2713                            <1>     ; 17/04/2021 - TRDOS 386 v2.0.4
2714                            <1>     ; 24/06/2016
2715                            <1>     ; 12/05/2016 - TRDOS 386 (TRDOS v2.0)
2716                            <1>     ; 27/06/2015
2717                            <1>     ; 02/09/2014
2718                            <1>     ; 30/08/2014 (Retro UNIX 386 v1)
2719                            <1>     ; 04/12/2013 (Retro UNIX 8086 v1)
2720                            <1>     ;
2721                            <1>     ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
2722                            <1>     ;
2723                            <1> ;-----------------------------------------
2724                            <1> ; POSITION
2725                            <1> ;   THIS SERVICE ROUTINE CALCULATES THE REGEN BUFFER ADDRESS
2726                            <1> ;   OF A CHARACTER IN THE ALPHA MODE
2727                            <1> ; INPUT
2728                            <1> ;   AX = ROW, COLUMN POSITION
2729                            <1> ; OUTPUT
2730                            <1> ;   AX = OFFSET OF CHAR POSITION IN REGEN BUFFER
2731                            <1> ;-----------------------------------------
2732                            <1>
2733                            <1>     ; DX = ROW, COLUMN POSITION
2734                            <1>     ;movzx  eax, byte [CRT_COLS] ; 27/06/2015
2735 00001FE1 31C0              <1>     xor     eax, eax ; 02/09/2014
2736 00001FE3 B050              <1>     mov     al, 80   ; determine bytes to row
2737 00001FE5 F6E6              <1>     mul     dh       ; row value
2738                            <1>     ;xor    dh, dh   ; 0
2739                            <1>     ;add    ax, dx   ; add column value to the result
2740                            <1>     ; 16/04/2021
2741 00001FE7 00D0              <1>     add     al, dl
2742 00001FE9 80D400            <1>     adc     ah, 0
2743                            <1>     ; 02/08/2022
2744 00001FEC D1E0              <1>     shl     eax, 1
2745                            <1>     ;shl    ax, 1   ; * 2 for attribute bytes
2746                            <1>             ; EAX = AX = OFFSET OF CHAR POSITION IN REGEN BUFFER
2747 00001FEE C3                <1>     retn
2748                            <1>
2749                            <1> find_position:
2750                            <1>     ; 17/04/2021 - TRDOS 386 v2.0.4
2751                            <1>     ; 24/06/2016
2752                            <1>     ; 12/05/2016 - TRDOS 386 (TRDOS v2.0)
2753                            <1>     ; 27/06/2015
2754                            <1>     ; 07/09/2014
2755                            <1>     ; 02/09/2014
2756                            <1>     ; 30/08/2014 (Retro UNIX 386 v1)
2757                            <1>     ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
2758                            <1>
2759 00001FEF 0FB6CF            <1>     movzx   ecx, bh ; video page number
2760                            <1>     ; 17/04/2021
2761                            <1>     ;mov    esi, ecx
2762                            <1>     ;shl    si, 1
2763                            <1>     ;mov    dx, [esi+CURSOR_POSN]
2764                            <1>     ;jz     short p21
2765                            <1>     ;xor    si, si
2766                            <1>     ; 17/04/2021
2767 00001FF2 31F6              <1>     xor     esi, esi
2768 00001FF4 D0E1              <1>     shl     cl, 1
2769 00001FF6 668B91[9E760100]  <1>     mov     dx, [ecx+CURSOR_POSN]
2770 00001FFD 740B              <1>     jz      short p21
2771 00001FFF D0E9              <1>     shr     cl, 1
2772                            <1> p20:
2773 00002001 660335[18830100]  <1>     add     si, [CRT_LEN] ; 24/06/2016
2774                            <1>     ;add    si, 80*25*2 ; add length of buffer for one page
2775 00002008 E2F7              <1>     loop    p20
2776                            <1> p21:
2777 0000200A 6621D2            <1>     and     dx, dx
```

```
2778 0000200D 7407              <1>     jz      short p22
2779 0000200F E8CDFFFFFF        <1>     call    position ; determine location in regen in page
2780 00002014 01C6              <1>     add     esi, eax ; add location to start of regen page
2781                            <1> p22:
2782                            <1>     ;mov    dx, [addr_6845] ; get base address of active display
2783                            <1>     ;mov    dx, 03D4h ; I/O address of color card
2784                            <1>     ;add    dx, 6   ; point at status port
2785 00002016 66BADA03          <1>     mov     dx, 03DAh ; status port
2786                            <1>     ; cx = 0
2787 0000201A C3                <1>     retn
2788                            <1>
2789                            <1> SCROLL_UP:
2790                            <1>     ; 04/08/2022 (TRDOS 386 v2.0.5)
2791                            <1>     ; 07/07/2016
2792                            <1>     ; 26/06/2016
2793                            <1>     ; 12/05/2016
2794                            <1>     ; 30/01/2016
2795                            <1>     ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
2796                            <1>     ; 07/09/2014
2797                            <1>     ; 02/09/2014
2798                            <1>     ; 01/09/2014 (Retro UNIX 386 v1 - beginning)
2799                            <1>     ; 04/04/2014
2800                            <1>     ; 04/12/2013
2801                            <1>     ;
2802                            <1>     ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
2803                            <1>     ;
2804                            <1> ;--------------------------------------------
2805                            <1> ; SCROLL UP
2806                            <1> ;   THIS ROUTINE MOVES A BLOCK OF CHARACTERS UP
2807                            <1> ;   ON THE SCREEN
2808                            <1> ; INPUT
2809                            <1> ;   (AH) = CURRENT CRT MODE
2810                            <1> ;   (AL) = NUMBER OF ROWS TO SCROLL
2811                            <1> ;   (CX) = ROW/COLUMN OF UPPER LEFT CORNER
2812                            <1> ;   (DX) = ROW/COLUMN OF LOWER RIGHT CORNER
2813                            <1> ;   (BH) = ATTRIBUTE TO BE USED ON BLANKED LINE
2814                            <1> ;   (DS) = DATA SEGMENT
2815                            <1> ;   (ES) = REGEN BUFFER SEGMENT
2816                            <1> ; OUTPUT
2817                            <1> ;   NONE -- THE REGEN BUFFER IS MODIFIED
2818                            <1> ;--------------------------------------------
2819                            <1>
2820                            <1>     ; 07/07/2016
2821 0000201B 38F5              <1>     cmp     ch, dh
2822                            <1>     ;ja     VIDEO_RETURN
2823                            <1>     ; 04/08/2022
2824 0000201D 7709              <1>     ja      short _s_u_retn
2825                            <1>
2826 0000201F 38D1              <1>     cmp     cl, dl
2827                            <1>     ;ja     VIDEO_RETURN
2828                            <1>     ; 04/08/2022
2829 00002021 7705              <1>     ja      short _s_u_retn
2830                            <1>     ;
2831 00002023 E805000000        <1>     call    _scroll_up
2832                            <1> _s_u_retn:
2833 00002028 E94BFBFFFF        <1>         jmp     VIDEO_RETURN
2834                            <1>
2835                            <1> _scroll_up:  ; from 'write_tty'
2836                            <1>     ;
2837                            <1>     ; cl = left upper column
2838                            <1>     ; ch = left upper row
2839                            <1>     ; dl = right lower column
2840                            <1>     ; dh = right lower row
2841                            <1>     ;
2842                            <1>     ; al = line count
2843                            <1>     ; bl = attribute to be used on blanked line
2844                            <1>     ; bh = video page number (0 to 7)
2845                            <1>
2846 0000202D E89B000000        <1>     call    test_line_count ; 16/01/2016
2847                            <1>
2848 00002032 8A25[1E680000]    <1>     mov     ah, [CRT_MODE] ; current video mode
2849                            <1>     ;;cmp   byte [CRT_MODE], 4
2850                            <1>     ;cmp    ah, 4 ; 07/07/2016
2851                            <1>     ;jnb    GRAPHICS_UP ; 26/06/2016
2852                            <1>     ; 18/11/2020
2853 00002038 80FC04            <1>     cmp     ah, 4
2854 0000203B 720A              <1>     jb      short n0
2855 0000203D 80FC07            <1>     cmp     ah, 7 ; TEST FOR BW CARD
2856                            <1>                   ; (80x25 text, mono)
2857 00002040 7405              <1>     je      short n0 ; same with mode 3 for TRDOS 386
2858 00002042 E91B050000        <1>     jmp     GRAPHICS_UP
2859                            <1> n0:
2860                            <1>     ; 07/07/2016
2861 00002047 80FF07            <1>     cmp     bh, 7 ; video page number
2862 0000204A 7606              <1>     jna     short n1
2863 0000204C 8A3D[AE760100]    <1>     mov     bh, [ACTIVE_PAGE]
2864                            <1> n1:
2865 00002052 88DC              <1>     mov     ah, bl ; attribute
2866                            <1>     ;push   ax ; *
2867                            <1>     ; 12/04/2021
2868 00002054 50                <1>     push    eax ; *
2869                            <1>     ;mov    esi, [CRT_BASE]
2870 00002055 BE00800B00        <1>         mov     esi, 0B8000h
2871 0000205A 3A3D[AE760100]    <1>         cmp     bh, [ACTIVE_PAGE]
2872 00002060 750B              <1>     jne     short n2
2873                            <1>     ;
2874 00002062 66A1[9C760100]    <1>         mov     ax, [CRT_START]
2875 00002068 6601C6            <1>         add     si, ax
2876 0000206B EB11              <1>         jmp     short n4
2877                            <1> n2:
2878 0000206D 20FF              <1>         and     bh, bh
2879 0000206F 740D              <1>     jz      short n4
2880 00002071 88F8              <1>     mov     al, bh
2881                            <1> n3:
2882 00002073 6603B5[18830100]  <1>         add si, [CRT_LEN]
2883 0000207A FEC8              <1>         dec al
2884 0000207C 75F5              <1>     jnz     short n3
2885                            <1> n4:
2886 0000207E E85B000000        <1>     call    scroll_position ; 16/01/2016
2887 00002083 7420              <1>         jz      short n6
2888                            <1>
2889 00002085 01CE              <1>         add     esi, ecx ; from address for scroll
2890 00002087 88F5              <1>     mov     ch, dh  ; #rows in block
2891 00002089 28C5              <1>     sub     ch, al  ; #rows to be moved
2892                            <1> n5:
2893 0000208B E88A000000        <1>     call    n10 ; 16/01/2016
2894                            <1>
2895 00002090 51                <1>         push       ecx
2896 00002091 0FB60D[20680000]  <1>     movzx   ecx, byte [CRT_COLS]
2897 00002098 00C9              <1>     add     cl, cl
2898 0000209A 01CE              <1>         add esi, ecx  ; next line
2899 0000209C 01CF              <1>         add edi, ecx
2900 0000209E 59                <1>     pop     ecx
2901                            <1>
```

```
0000209F FECD                <1>     dec     ch      ; count of lines to move
000020A1 75E8                <1>     jnz     short n5 ; row loop
                             <1>     ; ch = 0
000020A3 88C6                <1>     mov     dh, al   ; #rows
                             <1> n6:
                             <1>     ; attribute in ah
000020A5 B020                <1>     mov     al, ' '  ; fill with blanks
                             <1> n7:
000020A7 E87B000000          <1>     call    n11 ; 16/01/2016
                             <1>
000020AC 8A0D[20680000]      <1>     mov     cl, [CRT_COLS]
000020B2 00C9                <1>     add     cl, cl
000020B4 01CF                <1>         add edi, ecx
                             <1>
000020B6 FECE                <1>     dec     dh
000020B8 75ED                <1>     jnz     short n7
                             <1> n16:
000020BA 3A3D[AE760100]      <1>     cmp     bh, [ACTIVE_PAGE]
000020C0 750A                <1>     jne     short n8
                             <1>
                             <1>     ;cmp    byte [CRT_MODE], 7 ; is this the black and white card
                             <1>     ;je     short n8           ; if so, skip the mode reset
                             <1>
000020C2 A0[1F680000]        <1>     mov     al, [CRT_MODE_SET] ; get the value of mode set
000020C7 66BAD803            <1>     mov     dx, 03D8h ; always set color card port
000020CB EE                  <1>     out     dx, al
                             <1> n8:
000020CC C3                  <1>     retn
                             <1>
                             <1> test_line_count:
                             <1>     ; 12/04/2021
                             <1>     ; 12/05/2016
                             <1>     ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                             <1>     ; 07/09/2014 (scroll_up)
000020CD 08C0                <1>     or      al, al
000020CF 740C                <1>     jz      short al_set2
                             <1>     ;push   dx
                             <1>     ; 12/04/2021
000020D1 52                  <1>     push    edx
000020D2 28EE                <1>     sub     dh, ch  ; subtract upper row from lower row number
000020D4 FEC6                <1>     inc     dh      ; adjust difference by 1
000020D6 38C6                <1>     cmp     dh, al  ; line count = amount of rows in window?
000020D8 7502                <1>     jne     short al_set1 ; if not the we're all set
000020DA 30C0                <1>     xor     al, al  ; otherwise set al to zero
                             <1> al_set1:
                             <1>     ;pop    dx
                             <1>     ; 12/04/2021
000020DC 5A                  <1>     pop     edx
                             <1> al_set2:
000020DD C3                  <1>     retn
                             <1>
                             <1> scroll_position:
                             <1>     ; 12/04/2021
                             <1>     ; 26/06/2016
                             <1>     ; 30/01/2016
                             <1>         ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                             <1>     ; 07/09/2014 (scroll_up)
                             <1>
                             <1>     ; (*) [esp+4] = ax (al = line count, ah = attribute)
                             <1>
                             <1>     ;push   dx
                             <1>     ; 12/04/2021
000020DE 52                  <1>     push    edx
000020DF 6689CA              <1>     mov     dx, cx  ; now, upper left position in DX
000020E2 E8FAFEFFFF          <1>     call    position
000020E7 01C6                <1>     add     esi, eax
000020E9 89F7                <1>     mov     edi, esi
                             <1>     ;pop    dx      ; lower right position in DX
                             <1>     ; 12/04/2021
000020EB 5A                  <1>     pop     edx
000020EC 6629CA              <1>     sub     dx, cx
000020EF FEC6                <1>     inc     dh      ; dh = #rows
000020F1 FEC2                <1>     inc     dl      ; dl = #cols in block
000020F3 59                  <1>     pop     ecx     ; return address
                             <1>     ;pop    ax      ; * ; al = line count, ah = attribute
                             <1>     ; 12/04/2021
000020F4 58                  <1>     pop     eax ; (*)
000020F5 51                  <1>     push    ecx     ; return address
000020F6 0FB7C8              <1>     movzx   ecx, ax
000020F9 8A25[20680000]      <1>     mov     ah, [CRT_COLS]
000020FF F6E4                <1>     mul     ah      ; determine offset to from address
                             <1>     ;add    ax, ax  ; *2 for attribute byte
                             <1>     ; 02/08/2022
                             <1>     ;shl    eax, 1
00002101 01C0                <1>     add     eax, eax
                             <1>     ;
                             <1>     ;push   ax      ; offset
                             <1>     ;push   dx
                             <1>     ; 12/04/2021
00002103 50                  <1>     push    eax     ; offset
00002104 52                  <1>     push    edx
                             <1>     ;
                             <1>     ; 04/04/2014
00002105 66BADA03            <1>     mov     dx, 3DAh ; guaranteed to be color card here
                             <1> n9:                      ; wait_display_enable
00002109 EC                  <1>         in      al, dx   ; get port
0000210A A808                <1>     test    al, RVRT ; wait for vertical retrace
0000210C 74FB                <1>     jz      short n9 ; wait_display_enable
0000210E B025                <1>     mov     al, 25h
00002110 B2D8                <1>     mov     dl, 0D8h ; address control port
00002112 EE                  <1>     out     dx, al  ; turn off video during vertical retrace
                             <1>     ;pop    dx      ; #rows, #cols
                             <1>             ;pop    ax      ; offset
                             <1>     ; 12/04/2021
00002113 5A                  <1>     pop     edx     ; #rows, #cols
00002114 58                  <1>             pop     eax     ; offset
00002115 6691                <1>     xchg    ax, cx  ;
                             <1>     ; ecx = offset, al = line count, ah = attribute
                             <1>     ;
00002117 08C0                <1>     or      al, al
00002119 C3                  <1>     retn
                             <1> n10:
                             <1>     ; Move rows
0000211A 88D1                <1>     mov     cl, dl  ; get # of cols to move
0000211C 56                  <1>     push    esi
0000211D 57                  <1>     push    edi     ; save start address
                             <1> n10r:
0000211E 66A5                <1>     movsw           ; move that line on screen
00002120 FEC9                <1>     dec     cl
00002122 75FA                <1>         jnz     short n10r
00002124 5F                  <1>     pop     edi
00002125 5E                  <1>     pop     esi     ; recover addresses
00002126 C3                  <1>     retn
                             <1> n11:
```

```
                                <1>     ; Clear rows
                                <1>                     ; dh =  #rows
00002127 88D1                   <1>         mov cl, dl  ; get # of cols to clear
00002129 57                     <1>         push    edi     ; save address
                                <1> n11r:
0000212A 66AB                   <1>         stosw           ; store fill character
0000212C FEC9                   <1>     dec     cl
0000212E 75FA                   <1>         jnz     short n11r
00002130 5F                     <1>         pop     edi     ; recover address
00002131 C3                     <1>     retn
                                <1>
                                <1> SCROLL_DOWN:
                                <1>     ; 12/04/2021
                                <1>     ; 07/07/2016
                                <1>     ; 27/06/2016
                                <1>     ; 26/06/2016
                                <1>     ; 12/05/2016
                                <1>     ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>     ;
                                <1>     ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
                                <1>
                                <1> ;-----------------------------------------
                                <1> ; SCROLL DOWN
                                <1> ;   THIS ROUTINE MOVES THE CHARACTERS WITHIN A DEFINED
                                <1> ;   BLOCK DOWN ON THE SCREEN, FILLING THE TOP LINES
                                <1> ;   WITH A DEFINED CHARACTER
                                <1> ; INPUT
                                <1> ;   (AH) = CURRENT CRT MODE
                                <1> ;   (AL) = NUMBER OF LINES TO SCROLL
                                <1> ;   (CX) = UPPER LEFT CORNER OF RECION
                                <1> ;   (DX) = LOWER RIGHT CORNER OF REGION
                                <1> ;   (BH) = FILL CHARACTER
                                <1> ;   (DS) = DATA SEGMENT
                                <1> ;   (ES) = REGEN SEGMENT
                                <1> ; OUTPUT
                                <1> ;   NONE -- SCREEN IS SCROLLED
                                <1> ;-----------------------------------------
                                <1>
00002132 38F5                   <1>     cmp     ch, dh
                                <1>     ;ja     VIDEO_RETURN
00002134 7709                   <1>     ja      short _s_d_retn ; 18/11/2020
00002136 38D1                   <1>     cmp     cl, dl
                                <1>     ;ja     VIDEO_RETURN
00002138 7705                   <1>     ja      short _s_d_retn ; 18/11/2020
                                <1>     ;
0000213A E805000000             <1>     call    _scroll_down
                                <1> _s_d_retn:
0000213F E934FAFFFF             <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>     ; !!!!
00002144 FD                     <1>     std             ; DIRECTION FOR SCROLL DOWN
                                <1>     ; !!!!
00002145 E883FFFFFF             <1>     call    test_line_count ; 16/01/2016
                                <1>
0000214A 8A25[1E680000]         <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
00002150 80FC04                 <1>     cmp     ah, 4
00002153 720A                   <1>     jb      short _n0
00002155 80FC07                 <1>     cmp     ah, 7 ; TEST FOR BW CARD
                                <1>                   ;(80x25 text, mono)
00002158 7405                   <1>     je      short _n0 ; same with mode 3 for TRDOS 386
0000215A E9D5060000             <1>     jmp     GRAPHICS_DOWN
                                <1> _n0:
                                <1>     ; 07/07/2016
0000215F 80FF07                 <1>     cmp     bh, 7 ; video page number
00002162 7606                   <1>     jna     short n12
00002164 8A3D[AE760100]         <1>     mov     bh, [ACTIVE_PAGE]
                                <1>     ;
                                <1> n12:                ; CONTINUE_DOWN
0000216A 88DC                   <1>     mov     ah, bl
                                <1>     ;push   ax      ; * ; save attribute in ah
                                <1>     ; 12/04/2021
0000216C 50                     <1>     push    eax
0000216D 6689D0                 <1>     mov     ax, dx  ; LOWER RIGHT CORNER
00002170 E869FFFFFF             <1>     call    scroll_position       ; GET REGEN LOCATION
00002175 741F                   <1>     jz      short n14
00002177 29CE                   <1>     sub     esi, ecx  ; SI IS FROM ADDRESS
00002179 88F5                   <1>     mov     ch, dh  ; #rows in block
0000217B 28C5                   <1>     sub     ch, al  ; #rows to be moved
                                <1> n13:
0000217D E898FFFFFF             <1>     call    n10     ; MOVE ONE ROW
                                <1>
00002182 51                     <1>     push    ecx
00002183 8A0D[20680000]         <1>     mov     cl, [CRT_COLS]
00002189 00C9                   <1>     add     cl, cl
0000218B 29CE                   <1>         sub esi, ecx  ; next line
0000218D 29CF                   <1>         sub edi, ecx
0000218F 59                     <1>         pop ecx
                                <1>
00002190 FECD                   <1>     dec     ch      ; count of lines to move
00002192 75E9                   <1>     jnz     short n13 ; row loop
                                <1>     ; ch = 0
00002194 88C6                   <1>     mov     dh, al  ; #rows
                                <1> n14:
                                <1>     ; attribute in ah
00002196 B020                   <1>     mov     al, ' '  ; fill with blanks
                                <1> n15:
00002198 E88AFFFFFF             <1>     call    n11 ; 16/01/2016
                                <1>
0000219D 8A0D[20680000]         <1>     mov     cl, [CRT_COLS]
000021A3 00C9                   <1>     add     cl, cl
000021A5 29CF                   <1>         sub edi, ecx
                                <1>
000021A7 FECE                   <1>     dec     dh
000021A9 75ED                   <1>     jnz     short n15
                                <1>     ;
                                <1>     ; 18/11/2020
000021AB FC                     <1>     cld     ; clear direction flag
                                <1>     ;
```

```
3150 000021AC E909FFFFFF          <1>     jmp     n16 ; 27/06/2016
3151                              <1>
3152                              <1> ;   cmp     bh, [ACTIVE_PAGE]
3153                              <1> ;   jne     short n16
3154                              <1> ;
3155                              <1> ;   ;cmp    byte [CRT_MODE], 7    ; is this the black and white card
3156                              <1> ;   ;je     short n16             ; if so, skip the mode reset
3157                              <1> ;
3158                              <1> ;   mov     al, [CRT_MODE_SET] ; get the value of mode set
3159                              <1> ;   mov     dx, 03D8h ; always set color card port
3160                              <1> ;   out     dx, al
3161                              <1> ;n16:
3162                              <1> ;   ; !!!!
3163                              <1> ;   cld           ; Clear direction flag !
3164                              <1> ;   ; !!!!
3165                              <1> ;   retn
3166                              <1>
3167                              <1> READ_AC_CURRENT:
3168                              <1>     ; 08/07/2016
3169                              <1>     ; 26/06/2016
3170                              <1>     ; 12/05/2016
3171                              <1>     ; 18/01/2016
3172                              <1>     ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
3173                              <1>     ;
3174                              <1>     ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
3175                              <1>     ;
3176                              <1>     ; 08/07/2016
3177 000021B1 803D[1E680000]07    <1>        cmp     byte [CRT_MODE], 7 ; 6!?
3178 000021B8 7607                <1>     jna     short read_ac_c
3179 000021BA 31C0                <1>     xor     eax, eax
3180 000021BC E9BCF9FFFF          <1>        jmp     _video_return
3181                              <1> read_ac_c:
3182 000021C1 E805000000          <1>     call    _read_ac_current
3183                              <1>     ; 12/05/2016
3184                              <1>        ;jmp     VIDEO_RETURN
3185 000021C6 E9B2F9FFFF          <1>     jmp     _video_return
3186                              <1>
3187                              <1> ;--------------------------------------------------------------------------
3188                              <1> ; READ_AC_CURRENT                                                   :
3189                              <1> ;   THIS ROUTINE READS THE ATTRIBUTE AND CHARACTER AT THE CURRENT     :
3190                              <1> ;   CURSOR POSITION AND RETURNS THEM TO THE CALLER                   :
3191                              <1> ; INPUT                                                             :
3192                              <1> ;   (AH) = CURRENT CRT MODE                                         :
3193                              <1> ;   (BH) = DISPLAY PAGE ( ALPHA MODES ONLY )                    :
3194                              <1> ;   (DS) = DATA SEGMENT                                         :
3195                              <1> ;   (ES) = REGEN SEGMENT                                        :
3196                              <1> ; OUTPUT                                                            :
3197                              <1> ;   (AL) = CHARACTER READ                                       :
3198                              <1> ;   (AH) = ATTRIBUTE READ                                       :
3199                              <1> ;--------------------------------------------------------------------------
3200                              <1>
3201                              <1> _read_ac_current:
3202                              <1>     ; 26/06/2016
3203                              <1>     ; 12/05/2016
3204                              <1>     ; 18/01/2016
3205                              <1>
3206 000021CB 8A25[1E680000]      <1>     mov     ah, [CRT_MODE] ; current video mode
3207 000021D1 80FC04              <1>     cmp     ah, 4
3208 000021D4 720A                <1>     jb      short p10
3209                              <1>     ; 18/11/2020
3210                              <1>     ;cmp    byte [CRT_MODE], 4
3211                              <1>     ;jnb    GRAPHICS_READ ; 26/06/2016
3212                              <1>
3213 000021D6 80FC07              <1>     cmp     ah, 7 ; TEST FOR BW CARD (80x25 monochrome text)
3214 000021D9 7405                <1>     je      short p10       ; same with mode 3 in TRDOS 386
3215 000021DB E9A0080000          <1>     jmp     GRAPHICS_READ
3216                              <1> p10:
3217 000021E0 E80AFEFFFF          <1>     call    find_position  ; GET REGEN LOCATION AND PORT ADDRESS
3218                              <1>     ;
3219                              <1>     ; esi = regen location
3220                              <1>     ; dx = status port
3221                              <1>     ;
3222                              <1>
3223                              <1>     ; 18/11/2020
3224                              <1>     ; convert display mode to a zero value
3225                              <1>     ; for 80 column color mode
3226                              <1>     ;mov    ah, [CRT_MODE]
3227                              <1>     ;sub    ah, 2
3228                              <1>     ;shr    ah, 1
3229                              <1>     ;jnz    short p13
3230                              <1>
3231                              <1>     ; 05/12/2020
3232                              <1>     ; 18/11/2020 (TRDOS 386 v2.0.3)
3233                              <1>     ;xor    bl, bl  ; 0
3234 000021E5 803D[1E680000]03    <1>     cmp     byte [CRT_MODE], 03h ; 80x25 color text
3235                              <1>     ;je     short p11    ; Note: Only mode 03h and mode 01h are
3236                              <1>     ;inc    bl      ; 1  ;       in use by TRDOS 386 as text modes
3237                              <1>     ;jmp    short p14    ;       (07h, 00h and 02h are redirected)
3238 000021EC 7516                <1>     jne     short p13
3239                              <1>
3240                              <1>     ; 05/12/2020
3241 000021EE 3A3D[AE760100]      <1>     cmp     bh, [ACTIVE_PAGE]
3242 000021F4 750E                <1>     jne     short p13
3243                              <1>
3244                              <1>     ; WAIT FOR HORIZONTAL RETRACE OR VERTICAL RETRACE IF COLOR 80
3245                              <1> p11:
3246 000021F6 FB                  <1>     sti           ; enable interrupts first
3247                              <1>     ; 05/12/2020
3248 000021F7 90                  <1>     nop
3249                              <1>     ; 18/11/2020
3250                              <1>     ;or     bl, bl
3251                              <1>     ;jnz    short p13 ; mode 01h (and mode 00h) - 40x25 color text
3252 000021F8 FA                  <1>     cli           ; block interrupts for single loop
3253 000021F9 EC                  <1>     in      al, dx  ; get status from the adapter
3254 000021FA A801                <1>     test    al, RHRZ ; is horizontal retrace low
3255 000021FC 75F8                <1>     jnz     short p11 ; wait until it is
3256                              <1> p12:              ;  wait for either retrace high
3257 000021FE EC                  <1>     in      al, dx ; get status again
3258 000021FF A809                <1>     test    al, RVRT+RHRZ ; is horizontal or vertical retrace high
3259 00002201 74FB                <1>     jz      short p12 ; wait until either retrace active
3260                              <1> ;p14:
3261 00002203 FB                  <1>     sti
3262                              <1> p13:
3263 00002204 81C600800B00        <1>     add     esi, 0B8000h
3264 0000220A 668B06              <1>     mov     ax, [esi]
3265                              <1>
3266 0000220D C3                  <1>     retn    ; 18/01/2016
3267                              <1>
3268                              <1> WRITE_AC_CURRENT:
3269                              <1>     ; 08/07/2016
3270                              <1>     ; 26/06/2016
3271                              <1>     ; 24/06/2016
3272                              <1>     ; 12/05/2016
3273                              <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
0000220E 803D[1E680000]07   <1>     cmp     byte [CRT_MODE], 7 ; 6!?
00002215 760A               <1>     jna     short write_ac_c
                            <1>
00002217 E8B60A0000         <1>     call    vga_write_char_attr
0000221C E957F9FFFF         <1>     jmp     VIDEO_RETURN
                            <1>
                            <1> write_ac_c:
00002221 E834000000         <1>     call    _write_c_current
                            <1>
00002226 0FB6F7             <1>     movzx   esi, bh ; video page number (0 to 7)
00002229 889E[27680000]     <1>     mov     [esi+chr_attrib], bl ; color/attribute
                            <1>
0000222F E944F9FFFF         <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
00002234 803D[1E680000]07   <1>     cmp     byte [CRT_MODE], 7 ; 6!?
0000223B 760A               <1>     jna     short write_c_c
                            <1>
0000223D E8900A0000         <1>     call    vga_write_char_only
00002242 E931F9FFFF         <1>     jmp     VIDEO_RETURN
                            <1>
                            <1> write_c_c:
                            <1>     ;and    bh, 7 ; video page number (<= 7)
00002247 0FB6F7             <1>     movzx   esi, bh
0000224A 8A9E[27680000]     <1>     mov     bl, [esi+chr_attrib]
                            <1>
00002250 E805000000         <1>     call    _write_c_current
00002255 E91EF9FFFF         <1>         jmp     VIDEO_RETURN
                            <1>
                            <1> _write_c_current:  ; from 'write_tty'
                            <1>     ; 12/04/2021
                            <1>     ; 18/11/2020
                            <1>     ; 26/06/2016
                            <1>     ; 24/06/2016
                            <1>     ; 12/05/2016
                            <1>     ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                            <1>     ; 30/08/2014 (Retro UNIX 386 v1)
                            <1>     ; 18/01/2014
                            <1>     ; 04/12/2013
                            <1>     ;
                            <1>     ; VIDEO.ASM - 06/10/85 VIDEO DISPLAY BIOS
                            <1>
                            <1>     ; 18/11/2020
0000225A 8A25[1E680000]     <1>     mov     ah, [CRT_MODE] ; current video mode
00002260 80FC04             <1>     cmp     ah, 4
00002263 720A               <1>     jb      short p40
                            <1>
                            <1>     ;cmp    byte [CRT_MODE], 4
                            <1>         ;jnb       GRAPHICS_WRITE ; 26/06/2016
                            <1>
00002265 80FC07             <1>     cmp     ah, 7 ; TEST FOR BW CARD
00002268 7405               <1>     je      short p40
0000226A E963070000         <1>     jmp     GRAPHICS_WRITE
                            <1> p40:
                            <1>     ; al = character
                            <1>     ; bl = color/attribute
                            <1>     ; bh = video page
                            <1>     ; cx = count of characters to write
                            <1>     ;push   dx
                            <1>     ; 12/04/2021
0000226F 52                 <1>     push    edx ; *
00002270 88DC               <1>     mov     ah, bl  ; color/attribute (12/05/2016)
                            <1>     ;push   ax      ; save character & attribute/color
                            <1>     ;push   cx
                            <1>     ; 12/04/2021
00002272 50                 <1>     push    eax ; ** ; save character & attribute/color
00002273 51                 <1>     push    ecx ; ***
00002274 E876FDFFFF         <1>     call    find_position  ; get regen location and port address
                            <1>     ;pop    cx
                            <1>     ; 12/04/2021
00002279 59                 <1>     pop     ecx ; ***
                            <1>     ; esi = regen location
                            <1>     ; dx = status port
                            <1>     ;
0000227A 81C600800B00       <1>     add     esi, 0B8000h ; 30/08/2014 (crt_base)
                            <1>     ;
                            <1>     ; 18/11/2020
                            <1>     ; convert display mode to a zero value
                            <1>     ; for 80 column color mode
                            <1>     ;mov    ah, [CRT_MODE]
                            <1>     ;sub    ah, 2
```

```
                                <1>     ;shr    ah, 1
                                <1>     ;jnz    short p44    ; 26/06/2016
                                <1>
                                <1>     ; 05/12/2020
                                <1>     ; 18/11/2020 (TRDOS 386 v2.0.3)
                                <1>     ;xor    bl, bl  ; 0
00002280 803D[1E680000]03       <1>     cmp     byte [CRT_MODE], 03h ; 80x25 color text
                                <1>     ;je     short p41    ; Note: Only mode 03h and mode 01h are
                                <1>     ;inc    bl      ; 1  ;       in use by TRDOS 386 as text modes
                                <1>     ;jmp    short p43    ;       (07h, 00h and 02h are redirected)
                                <1>     ; 05/12/2020
00002287 751A                   <1>     jne     short p44
                                <1> p46:
                                <1>     ; 05/12/2020
00002289 3A3D[AE760100]         <1>     cmp     bh, [ACTIVE_PAGE]
0000228F 7512                   <1>     jne     short p44
                                <1>
                                <1>     ; WAIT FOR HORIZONTAL RETRACE OR VERTICAL RETRACE IF COLOR 80
                                <1> p41:
                                <1>     ; 05/12/2020
00002291 FB                     <1>     sti             ; enable interrupts first
00002292 90                     <1>     nop
                                <1>     ; 18/11/2020
                                <1>     ;or     bl, bl
                                <1>     ;jnz    short p44 ; mode 01h (and mode 00h) - 40x25 color text
00002293 FA                     <1>     cli             ; block interrupts for single loop
00002294 EC                     <1>     in      al, dx  ; get status from the adapter
00002295 A808                   <1>     test    al, RVRT ; check for vertical retrace first
00002297 7509                   <1>     jnz     short p43 ; Do fast write now if vertical retrace
00002299 A801                   <1>     test    al, RHRZ ; is horizontal retrace low
0000229B 75F4                   <1>     jnz     short p41 ; wait until it is
                                <1> p42:                ;  wait for either retrace high
0000229D EC                     <1>     in      al, dx ; get status again
0000229E A809                   <1>     test    al, RVRT+RHRZ ; is horizontal or vertical retrace high
000022A0 74FB                   <1>     jz      short p42 ; wait until either retrace active
                                <1> p43:
000022A2 FB                     <1>     sti
                                <1> p44:
000022A3 668B0424               <1>     mov     ax, [esp] ; restore the character (al) & attribute (ah)
000022A7 668906                 <1>     mov     [esi], ax
                                <1>
000022AA 6649                   <1>     dec     cx
000022AC 740D                   <1>     jz      short p45
                                <1>
000022AE 46                     <1>     inc     esi
000022AF 46                     <1>     inc     esi
                                <1>
                                <1>     ; 05/12/2020
000022B0 803D[1E680000]03       <1>     cmp     byte [CRT_MODE], 03h
000022B7 75EA                   <1>     jne     short p44
                                <1>     ;jmp    short p41
000022B9 EBCE                   <1>     jmp     short p46
                                <1> p45:
                                <1>     ;pop    ax
                                <1>     ;pop    dx
                                <1>     ; 12/04/2021
000022BB 58                     <1>     pop     eax ; **
000022BC 5A                     <1>     pop     edx ; *
000022BD C3                     <1>     retn
                                <1>
                                <1> ; 09/07/2016
                                <1> ; 26/06/2016
                                <1> ; 24/06/2016
                                <1> ; 12/05/2016
                                <1> ; 18/01/2016
                                <1> ; 16/01/2016 - TRDOS 386 (TRDOS v2.0)
                                <1> ; 30/06/2015
                                <1> ; 27/06/2015
                                <1> ; 11/03/2015
                                <1> ; 02/09/2014
                                <1> ; 30/08/2014
                                <1> ; VIDEO FUNCTIONS
                                <1> ; (write_tty - Retro UNIX 8086 v1 - U9.ASM, 01/02/2014)
                                <1>
                                <1> WRITE_TTY:
                                <1>     ; 09/12/2017
                                <1>     ; 09/07/2016
                                <1>     ; 01/07/2016
                                <1>     ; 26/06/2016
                                <1>     ; 24/06/2016
                                <1>     ; 13/05/2016
                                <1>     ; 12/05/2016
                                <1>     ; 30/01/2016
                                <1>     ; 18/01/2016
                                <1>     ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>     ; 13/08/2015
                                <1>     ; 02/09/2014
                                <1>     ; 30/08/2014 (Retro UNIX 386 v1 - beginning)
                                <1>     ; 01/02/2014 (Retro UNIX 8086 v1 - last update)
                                <1>     ; 03/12/2013 (Retro UNIX 8086 v1 - beginning)
                                <1>     ; (Modified registers: EAX, EBX, ECX, EDX, ESI, EDI)
                                <1>     ;
                                <1>     ; INPUT -> AL = Character to be written
                                <1>     ;          BL = Color (Forecolor, Backcolor)
                                <1>     ;          BH = Video Page (0 to 7)
                                <1>
                                <1>     ; 09/07/2016
000022BE 803D[1E680000]07       <1>        cmp     byte [CRT_MODE], 7
000022C5 760A                   <1>     jna     short write_tty_cga
                                <1>
000022C7 E8EB0C0000            <1>     call    vga_write_teletype
000022CC E9A7F8FFFF             <1>     jmp     VIDEO_RETURN
                                <1>
                                <1> write_tty_cga:
                                <1>     ; 13/05/2016
                                <1>     ;call  _write_tty
                                <1>     ; 01/07/2016
000022D1 E818000000             <1>     call    _write_tty_m3
000022D6 E99DF8FFFF             <1>     jmp     VIDEO_RETURN
                                <1>
                                <1> RVRTequ     00001000b       ; VIDEO VERTICAL RETRACE BIT
                                <1> RHRZequ     00000001b       ; VIDEO HORIZONTAL RETRACE BIT
                                <1>
                                <1> ; Derived from "WRITE_TTY" procedure of IBM "pc-at" rombios source code
                                <1> ; (06/10/1985), 'video.asm', INT 10H, VIDEO_IO
                                <1> ;
                                <1> ; 06/10/85  VIDEO DISPLAY BIOS
                                <1> ;
                                <1> ;--- WRITE_TTY ----------------------------------------------------------------
                                <1> ;                                                                           :
                                <1> ;   THIS INTERFACE PROVIDES A TELETYPE LIKE INTERFACE TO THE                :
                                <1> ;   VIDEO CARDS. THE INPUT CHARACTER IS WRITTEN TO THE CURRENT               :
                                <1> ;   CURSOR POSITION, AND THE CURSOR IS MOVED TO THE NEXT POSITION.          :
                                <1> ;   IF THE CURSOR LEAVES THE LAST COLUMN OF THE FIELD, THE COLUMN           :
```

```
                            <1> ;   IS SET TO ZERO, AND THE ROW VALUE IS INCREMENTED. IF THE ROW             :
                            <1> ;   ROW VALUE LEAVES THE FIELD, THE CURSOR IS PLACED ON THE LAST ROW,        :
                            <1> ;   FIRST COLUMN, AND THE ENTIRE SCREEN IS SCROLLED UP ONE LINE.             :
                            <1> ;   WHEN THE SCREEN IS SCROLLED UP, THE ATTRIBUTE FOR FILLING THE            :
                            <1> ;   NEWLY BLANKED LINE IS READ FROM THE CURSOR POSITION ON THE PREVIOUS              :
                            <1> ;   LINE BEFORE THE SCROLL, IN CHARACTER MODE. IN GRAPHICS MODE,             :
                            <1> ;   THE 0 COLOR IS USED.                                                    :
                            <1> ;   ENTRY --                                                                :
                            <1> ;     (AH) = CURRENT CRT MODE                                               :
                            <1> ;     (AL) = CHARACTER TO BE WRITTEN                                        :
                            <1> ;       NOTE THAT BACK SPACE, CARRIAGE RETURN, BELL AND LINE FEED ARE :
                            <1> ;       HANDLED AS COMMANDS RATHER THAN AS DISPLAY GRAPHICS CHARACTERS:
                            <1> ;     (BL) = FOREGROUND COLOR FOR CHAR WRITE IF CURRENTLY IN A GRAPHICS MODE  :
                            <1> ;   EXIT --                                                                 :
                            <1> ;     ALL REGISTERS SAVED                                                   :
                            <1> ;-----------------------------------------------------------------------------
                            <1>
                            <1> ; 02/08/2022 (TRDOS 386 v2.0.5)
                            <1> ; 18/11/2020 (TRDOS 386 v2.0.3)
                            <1> ; 09/12/2017
                            <1> ; 08/07/2016
                            <1> ; 26/06/2016
                            <1> ; 24/06/2016
                            <1> _write_tty:
                            <1>     ; 13/05/2016
                            <1>     ; --- 18/11/2020 ---
                            <1>     ; NOTE:
                            <1>     ; Only kernel calls "_write_tty" procedure...
                            <1>     ; TRDOS 386 v2 kernel uses video mode 3 for displaying
                            <1>     ; (some error) messages and also mainprog command interpreter
                            <1>     ; (in kernel) uses "_write_tty".
                            <1>     ; So, here video mode must be set to 3 if it is not 3.
                            <1>
000022DB FA                 <1>     cli     ; disable interrupts
                            <1>     ;
                            <1>     ; 01/09/2014
000022DC 803D[1E680000]03   <1>     cmp     byte [CRT_MODE], 3
000022E3 7409               <1>     je      short _write_tty_m3
                            <1>     ;
                            <1> set_mode_3:
000022E5 53                 <1>     push    ebx
000022E6 50                 <1>     push    eax
                            <1>     ;call   _set_mode
                            <1>     ; 18/11/2020
000022E7 E89BF8FFFF         <1>     call    set_txt_mode  ; set video mode to 03h
000022EC 58                 <1>     pop     eax
000022ED 5B                 <1>     pop     ebx
                            <1>     ;
                            <1> _write_tty_m3: ; 24/06/2016 (m3 -> _write_tty_m3)
000022EE 0FB6F7             <1>     movzx   esi, bh ; 12/05/2016
                            <1>     ;shl    si, 1
                            <1>     ; 02/08/2022
000022F1 D1E6               <1>     shl     esi, 1
000022F3 81C6[9E760100]     <1>     add     esi, CURSOR_POSN
000022F9 668B16             <1>     mov     dx, [esi]
                            <1>     ;
                            <1>     ; dx now has the current cursor position
                            <1>     ;
000022FC 3C0D               <1>     cmp     al, 0Dh ; CR    ; is it carriage return or control character
                            <1>     ;jbe    short u8
                            <1>     ; 17/04/2021
000022FE 7241               <1>     jb      short u8
00002300 746F               <1>     je      short u9
                            <1>     ;
                            <1>     ; write the char to the screen
                            <1> u0:
                            <1>     ; al = character
                            <1>     ; bl = attribute/color
                            <1>     ; bh = video page number (0 to 7)
                            <1>     ;
                            <1>     ;mov    cx, 1  ; 24/06/2016
                            <1>     ; 02/08/2022
00002302 29C9               <1>     sub     ecx, ecx
00002304 FEC1               <1>     inc     cl ; ecx = 1
                            <1>     ; cx = count of characters to write
                            <1>     ;
00002306 E84FFFFFFF         <1>     call    _write_c_current ; 16/01/2015
                            <1>     ;
                            <1>     ; position the cursor for next char
0000230B FEC2               <1>     inc     dl              ; next column
0000230D 3A15[20680000]     <1>     cmp     dl, [CRT_COLS]  ; test for column overflow
                            <1>     ;jne    _set_cpos
                            <1>     ; 02/08/2022
00002313 7402               <1>     je      short u13
00002315 EB5C               <1>     jmp     _set_cpos
                            <1> u13:
00002317 B200               <1>     mov     dl, 0           ; column = 0
                            <1> u10:                        ; (line feed found)
00002319 80FE18             <1>     cmp     dh, 25-1        ; check for last row
0000231C 721F               <1>     jb      short u6
                            <1>     ;
                            <1>     ; scroll required
                            <1> u1:
                            <1>     ; SET CURSOR POSITION (04/12/2013)
0000231E E850000000         <1>     call    _set_cpos
                            <1>     ;
                            <1>     ; determine value to fill with during scroll
                            <1> u2:
                            <1>     ; bh = video page number
                            <1>     ;
00002323 E8A3FEFFFF         <1>     call    _read_ac_current ; 18/01/2016
                            <1>     ;
                            <1>     ; al = character, ah = attribute
                            <1>     ; bh = video page number
                            <1>     ; 18/11/2020
00002328 88E3               <1>     mov     bl, ah ; color/attribute
                            <1> u3:
                            <1>     ;;mov   ax, 0601h       ; scroll one line
                            <1>     ;;sub   cx, cx          ; upper left corner
                            <1>     ;;mov   dh, 25-1        ; lower right row
                            <1>     ;;;mov  dl, [CRT_COLS]
                            <1>     ;mov    dl, 80          ; lower right column
                            <1>     ;;dec   dl
                            <1>     ;;mov   dl, 79
                            <1>
                            <1>     ;;call  scroll_up       ; 04/12/2013
                            <1>     ;;; 11/03/2015
                            <1>     ; 02/09/2014
                            <1>     ;;;mov  cx, [crt_ulc] ; Upper left corner  (0000h)
                            <1>     ;;;mov  dx, [crt_lrc] ; Lower right corner (184Fh)
                            <1>     ; 11/03/2015
                            <1>     ;sub    cx, cx
                            <1>     ; 17/04/2021
0000232A 29C9               <1>     sub     ecx, ecx
```

```
                            <1>     ;mov    dx, 184Fh ; dl = 79 (column), dh = 24 (row)
                            <1>     ; 18/11/2020
0000232C B618               <1>     mov     dh, 25-1
0000232E 8A15[20680000]     <1>     mov     dl, [CRT_COLS]
00002334 FECA               <1>     dec     dl
                            <1>     ;
00002336 B001               <1>     mov     al, 1           ; scroll 1 line up
                            <1>             ; ah = attribute
                            <1>     ;mov    bl, al ; 12/05/2016
00002338 E9F0FCFFFF         <1>     jmp     _scroll_up      ; 16/01/2016
                            <1> ;u4:
                            <1>     ;;int   10h             ; video-call return
                            <1>                             ; scroll up the screen
                            <1>                             ; tty return
                            <1> ;u5:
                            <1>     ;retn                   ; return to the caller
                            <1>
                            <1> u6:                         ; set-cursor-inc
0000233D 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
0000233F EB32               <1>     jmp     _set_cpos
                            <1>
                            <1>     ; check for control characters
                            <1> u8:
                            <1>     ;je     short u9 ; 17/04/2021
00002341 3C0A               <1>     cmp     al, 0Ah         ; is it a line feed (0Ah)
00002343 74D4               <1>     je      short u10
00002345 3C07               <1>     cmp     al, 07h         ; is it a bell
00002347 747C               <1>     je      short u11
00002349 3C08               <1>     cmp     al, 08h         ; is it a backspace
                            <1>     ;jne    short u0
0000234B 741C               <1>     je      short bs        ; 12/12/2013
                            <1>     ; 12/12/2013 (tab stop)
0000234D 3C09               <1>     cmp     al, 09h         ; is it a tab stop
0000234F 75B1               <1>     jne     short u0
00002351 88D0               <1>     mov     al, dl
                            <1>     ;cbw
00002353 30E4               <1>     xor     ah, ah ; 09/12/2017
00002355 B108               <1>     mov     cl, 8
00002357 F6F1               <1>     div     cl
00002359 28E1               <1>     sub     cl, ah
                            <1> ts:
                            <1>     ; 02/09/2014
                            <1>     ; 01/09/2014
                            <1>     ;mov    al, 20h
                            <1> tsloop:
                            <1>     ;push   cx
                            <1>     ; 12/04/2021
0000235B 51                 <1>     push    ecx ; *
                            <1>     ; 18/11/2020
                            <1>     ;push   ax
                            <1>     ;;mov   bh, [ACTIVE_PAGE]
                            <1>     ; 05/12/2020
                            <1>     ;push   bx
0000235C B020               <1>     mov     al, 20h ; al = space (blank)
                            <1>                     ; bl = color/attribute
0000235E E88BFFFFFF         <1>     call    _write_tty_m3 ; 24/06/2016 (m3 -> _write_tty_m3)
                            <1>     ; 05/12/2020
                            <1>     ; bx is preserved in '_write_tty_m3'
                            <1>     ; 18/11/2020
                            <1>     ;pop    bx  ; BL = color/attribute, Bh = video page
                            <1>     ;pop    ax  ; AL = character
                            <1>     ; 12/04/2021
                            <1>     ;pop    cx
00002363 59                 <1>     pop     ecx ; *
00002364 FEC9               <1>     dec     cl
00002366 75F3               <1>     jnz     short tsloop
00002368 C3                 <1>     retn
                            <1> bs:
                            <1>     ; back space found
                            <1>
00002369 08D2               <1>     or      dl, dl          ; is it already at start of line
                            <1>     ;je     short u7        ; set_cursor
0000236B 7406               <1>     jz      short _set_cpos
                            <1>     ;dec    dx              ; no -- just move it back
                            <1>     ; 17/04/2021
0000236D FECA               <1>     dec     dl              ; move to 1 column back
                            <1>     ;jmp    short u7
0000236F EB02               <1>     jmp     short _set_cpos
                            <1>
                            <1>     ; carriage return found
                            <1> u9:
00002371 B200               <1>     mov     dl, 0           ; move to first column
                            <1>     ;jmp    short u7
                            <1>     ;jmp    short _set_cpos ; 30/01/2016
                            <1>
                            <1>     ; line feed found
                            <1> ;u10:
                            <1> ;   cmp     dh, 25-1        ; bottom of screen
                            <1> ;   jne     short u6        ; no, just set the cursor
                            <1> ;       jmp     u1              ; yes, scroll the screen
                            <1>
                            <1> _set_cpos:
                            <1>     ; 02/08/2022 - TRDOS 386 v2.0.5
                            <1>     ; 17/04/2021 - TRDOS 386 v2.0.4
                            <1>     ; 12/05/2016 - TRDOS 386 (TRDOS v2.0)
                            <1>     ; 27/06/2015
                            <1>     ; 01/09/2014
                            <1>     ; 30/08/2014 (Retro UNIX 386 v1)
                            <1>     ;
                            <1>     ; 04/12/2013 - 12/12/2013 (Retro UNIX 8086 v1)
                            <1>     ;
                            <1>     ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
                            <1>     ;
                            <1> ;-----------------------------------------------
                            <1> ; SET_CPOS
                            <1> ;   THIS ROUTINE SETS THE CURRENT CURSOR POSITION TO THE
                            <1> ;   NEW X-Y VALUES PASSED
                            <1> ; INPUT
                            <1> ;   DX - ROW,COLUMN OF NEW CURSOR
                            <1> ;   BH - DISPLAY PAGE OF CURSOR
                            <1> ; OUTPUT
                            <1> ;   CURSOR ID SET AT 6845 IF DISPLAY PAGE IS CURRENT DISPLAY
                            <1> ;-----------------------------------------------
                            <1>     ;
00002373 BE[9E760100]       <1>     mov     esi, CURSOR_POSN
00002378 0FB6C7             <1>         movzx   eax, bh    ; BH = video page number
                            <1> ;   or      al, al
                            <1> ;   jz      short _set_cpos_0
```

```
                        <1>         shl     al, 1   ; word offset
                        <1>         add     esi, eax
                        <1> ;_set_cpos_0:
0000237F 668916         <1>     mov     [esi], dx ; save the pointer
00002382 383D[AE760100] <1>     cmp     [ACTIVE_PAGE], bh
00002388 7532           <1>     jne     short m17
                        <1>     ;call   m18     ; CURSOR SET
                        <1> ;m17:                       ; SET_CPOS_RETURN
                        <1>     ; 01/09/2014
                        <1> ;   retn
                        <1>             ; DX = row/column
                        <1> m18:
0000238A E852FCFFFF     <1>     call    position ; determine location in regen buffer
0000238F 668B0D[9C760100] <1>   mov     cx, [CRT_START]
00002396 6601C1         <1>     add     cx, ax  ; add char position in regen buffer
                        <1>                     ; to the start address (offset) for this page
00002399 66D1E9         <1>     shr     cx, 1   ; divide by 2 for char only count
0000239C B40E           <1>     mov     ah, 14  ; register number for cursor
                        <1>     ;call   m16     ; output value to the 6845
                        <1>     ;retn
                        <1>
                        <1>     ;-----  THIS ROUTINE OUTPUTS THE CX REGISTER
                        <1>     ;       TO THE 6845 REGISTERS NAMED IN (AH)
                        <1> m16:
0000239E FA             <1>     cli
                        <1>     ;mov    dx, [addr_6845] ; address register
0000239F 66BAD403       <1>     mov     dx, 03D4h ; I/O address of color card
000023A3 88E0           <1>     mov     al, ah  ; get value
000023A5 EE             <1>     out     dx, al  ; register set
                        <1>     ;inc    dx      ; data register
                        <1>     ; 17/04/2021
000023A6 FEC2           <1>     inc     dl
000023A8 EB00           <1>     jmp     $+2     ; i/o delay
000023AA 88E8           <1>     mov     al, ch  ; data
000023AC EE             <1>     out     dx, al
                        <1>     ;dec    dx
                        <1>     ; 17/04/2021
000023AD FECA           <1>     dec     dl
000023AF 88E0           <1>     mov     al, ah
000023B1 FEC0           <1>     inc     al      ; point to other data register
000023B3 EE             <1>     out     dx, al  ; set for second register
                        <1>     ;inc    dx
                        <1>     ; 17/04/2021
000023B4 FEC2           <1>     inc     dl
000023B6 EB00           <1>     jmp     $+2     ; i/o delay
000023B8 88C8           <1>     mov     al, cl  ; second data value
000023BA EE             <1>     out     dx, al
000023BB FB             <1>     sti
                        <1> m17:
000023BC C3             <1>     retn
                        <1>
                        <1> _beep:
                        <1>     ; 12/02/2021 (TRDOS v2.0.3)
000023BD FA             <1>     cli
000023BE E811000000     <1>     call    beep
000023C3 FB             <1>     sti
000023C4 C3             <1>     retn
                        <1>
                        <1> beeper:
                        <1>     ; 04/08/2016
                        <1>     ; 12/05/2016 - TRDOS 386 (TRDOS v2.0)
                        <1>     ; 30/08/2014 (Retro UNIX 386 v1)
                        <1>     ; 18/01/2014
                        <1>     ; 03/12/2013
                        <1>     ; bell found
                        <1> u11:
000023C5 FB             <1>     sti
000023C6 3A3D[AE760100] <1>     cmp     bh, [ACTIVE_PAGE]
000023CC 7552           <1>     jne     short u12       ; Do not sound the beep
                        <1>                             ; if it is not written on the active page
                        <1> beeper_gfx: ; 04/08/2016
000023CE 66B93305       <1>     mov     cx, 1331        ; divisor for 896 hz tone
000023D2 B31F           <1>     mov     bl, 31          ; set count for 31/64 second for beep
                        <1>     ;call   beep            ; sound the pod bell
                        <1>     ;jmp    short u5        ; tty_return
                        <1>     ;retn
                        <1>
                        <1> TIMER       equ     040h            ; 8254 TIMER - BASE ADDRESS
                        <1> PORT_B      equ     061h            ; PORT B READ/WRITE DIAGNOSTIC REGISTER
                        <1> GATE2       equ     00000001b       ; TIMER 2 INPUT CATE CLOCK BIT
                        <1> SPK2equ     00000010b       ; SPEAKER OUTPUT DATA ENABLE BIT
                        <1>
                        <1> beep:
                        <1>     ; 12/02/2021
                        <1>     ; 07/02/2015
                        <1>     ; 30/08/2014 (Retro UNIX 386 v1)
                        <1>     ; 18/01/2014
                        <1>     ; 03/12/2013
                        <1>     ;
                        <1>     ; TEST4.ASM - 06/10/85  POST AND BIOS UTILITY ROUTINES
                        <1>     ;
                        <1>     ; ROUTINE TO SOUND THE BEEPER USING TIMER 2 FOR TONE
                        <1>     ;
                        <1>     ; ENTRY:
                        <1>     ;    (BL) = DURATION COUNTER ( 1 FOR 1/64 SECOND )
                        <1>     ;    (CX) = FREQUENCY DIVISOR (1193180/FREQUENCY) (1331 FOR 886 HZ)
                        <1>     ; EXIT:                         :
                        <1>     ;    (AX),(BL),(CX) MODIFIED.
                        <1>
000023D4 9C             <1>     pushfd  ; 18/01/2014  ; save interrupt status
000023D5 FA             <1>     cli                   ; block interrupts during update
000023D6 B0B6           <1>     mov     al, 10110110b ; select timer 2, lsb, msb binary
000023D8 E643           <1>     out     TIMER+3, al   ; write timer mode register
000023DA EB00           <1>     jmp     $+2           ; I/O delay
000023DC 88C8           <1>     mov     al, cl        ; divisor for hz (low)
000023DE E642           <1>     out     TIMER+2,AL    ; write timer 2 count - lsb
000023E0 EB00           <1>     jmp     $+2           ; I/O delay
000023E2 88E8           <1>     mov     al, ch        ; divisor for hz (high)
000023E4 E642           <1>     out     TIMER+2, al   ; write timer 2 count - msb
000023E6 E461           <1>     in      al, PORT_B    ; get current setting of port
000023E8 88C4           <1>     mov     ah, al        ; save that setting
000023EA 0C03           <1>     or      al, GATE2+SPK2 ; gate timer 2 and turn speaker on
000023EC E661           <1>     out     PORT_B, al    ; and restore interrupt status
                        <1>     ; 12/02/2021
000023EE 9D             <1>     popfd  ; 18/01/2014
                        <1>     ;sti
                        <1> g7:                       ; 1/64 second per count (bl)
000023EF B90B040000     <1>     mov     ecx, 1035     ; delay count for 1/64 of a second
000023F4 E828000000     <1>     call    waitf         ; go to beep delay 1/64 count
000023F9 FECB           <1>     dec     bl            ; (bl) length count expired?
000023FB 75F2           <1>     jnz     short g7      ; no - continue beeping speaker
                        <1>     ;
000023FD 9C             <1>     pushfd ; 12/02/2021   ; save interrupt status
000023FE FA             <1>     cli    ; 18/01/2014   ; block interrupts during update
```

```
3894 000023FF E461              <1>     in      al, PORT_B      ; get current port value
3895                            <1>         ;or al, not (GATE2+SPK2) ; isolate current speaker bits in case
3896 00002401 0CFC              <1>         or       al, ~(GATE2+SPK2)
3897 00002403 20C4              <1>         and ah, al          ; someone turned them off during beep
3898 00002405 88E0              <1>     mov     al, ah          ; recover value of port
3899                            <1>         ;or al, not (GATE2+SPK2) ; force speaker data off
3900 00002407 0CFC              <1>     or      al, ~(GATE2+SPK2) ; isolate current speaker bits in case
3901 00002409 E661              <1>     out     PORT_B, al      ; and stop speaker timer
3902 0000240B 9D                <1>     popfd  ; 12/02/2021           ; restore interrupt flag state
3903                            <1>     ;sti
3904                            <1>     ;mov    ecx, 1035       ; force 1/64 second delay (short)
3905                            <1>     ; 17/04/2021
3906 0000240C 66B90B04          <1>     mov     cx, 1035
3907 00002410 E80C000000        <1>     call    waitf           ; minimum delay between all beeps
3908 00002415 9C                <1>     pushfd                  ; save interrupt status
3909 00002416 FA                <1>     cli                     ; block interrupts during update
3910 00002417 E461              <1>     in      al, PORT_B      ; get current port value in case
3911 00002419 2403              <1>     and     al, GATE2+SPK2 ; someone turned them on
3912 0000241B 08E0              <1>     or      al, ah          ; recover value of port_b
3913 0000241D E661              <1>     out     PORT_B, al      ; restore speaker status
3914 0000241F 9D                <1>     popfd                   ; restore interrupt flag state
3915                            <1> u12:
3916 00002420 C3                <1>     retn
3917                            <1>
3918                            <1> REFRESH_BIT equ    00010000b      ; REFRESH TEST BIT
3919                            <1>
3920                            <1> WAITF:
3921                            <1> waitf:
3922                            <1>     ; 30/08/2014 (Retro UNIX 386 v1)
3923                            <1>     ; 03/12/2013
3924                            <1>     ;
3925                            <1> ;   push    ax              ; save work register (ah)
3926                            <1> ;waitf1:
3927                            <1>                             ; use timer 1 output bits
3928                            <1> ;   in      al, PORT_B      ; read current counter output status
3929                            <1> ;   and     al, REFRESH_BIT       ; mask for refresh determine bit
3930                            <1> ;   cmp     al, ah          ; did it just change
3931                            <1> ;   je      short waitf1    ; wait for a change in output line
3932                            <1> ;   ;
3933                            <1> ;   mov     ah, al          ; save new lflag state
3934                            <1> ;   loop    waitf1          ; decrement half cycles till count end
3935                            <1> ;   ;
3936                            <1> ;   pop     ax              ; restore (ah)
3937                            <1> ;   retn                    ; return (cx)=0
3938                            <1>
3939                            <1> ; 06/02/2015 (unix386.s <-- dsectrm2.s)
3940                            <1> ; 17/12/2014 (dsectrm2.s)
3941                            <1> ; WAITF
3942                            <1> ; /// IBM PC-XT Model 286 System BIOS Source Code - Test 4 - 06/10/85 ///
3943                            <1> ;
3944                            <1> ;---WAITF----------------------------------------------------------------------
3945                            <1> ;   FIXED TIME WAIT ROUTINE (HARDWARE CONTROLLED - NOT PROCESSOR)
3946                            <1> ; ENTRY:
3947                            <1> ;   (CX) =  COUNT OF 15.085737 MICROSECOND INTERVALS TO WAIT
3948                            <1> ;           MEMORY REFRESH TIMER 1 OUTPUT USED AS REFERENCE
3949                            <1> ; EXIT:
3950                            <1> ;           AFTER (CX) TIME COUNT (PLUS OR MINUS 16 MICROSECONDS)
3951                            <1> ;   (CX) = 0
3952                            <1> ;-----------------------------------------------------------------------------
3953                            <1>
3954                            <1> ; Refresh period: 30 micro seconds (15-80 us)
3955                            <1> ; (16/12/2014 - AWARDBIOS 1999 - ATORGS.ASM, WAIT_REFRESH)
3956                            <1>
3957                            <1> ;WAITF:                                 ; DELAY FOR (CX)*15.085737 US
3958                            <1>     ;push   AX                      ; SAVE WORK REGISTER (AH)
3959                            <1>     ; 11/04/2021
3960 00002421 50                <1>     push    eax
3961                            <1>     ; 16/12/2014
3962                            <1>     ;shr    cx, 1                   ; convert to count of 30 micro seconds
3963 00002422 D1E9              <1>     shr     ecx, 1  ; 21/02/2015
3964                            <1> ;17/12/2014
3965                            <1> ;WAITF1:
3966                            <1> ;   in      al, PORT_B   ;061h     ; READ CURRENT COUNTER OUTPUT STATUS
3967                            <1> ;   and     al, REFRESH_BIT       ;00010000b ; MASK FOR REFRESH DETERMINE BIT
3968                            <1> ;   cmp     al, ah                  ; DID IT JUST CHANGE
3969                            <1> ;   je      short WAITF1            ; WAIT FOR A CHANGE IN OUTPUT LINE
3970                            <1> ;   mov     ah, al                  ; SAVE NEW FLAG STATE
3971                            <1> ;   loop    WAITF1                  ; DECREMENT HALF CYCLES TILL COUNT END
3972                            <1>     ;
3973                            <1>     ; 17/12/2014
3974                            <1>     ;
3975                            <1>     ; Modification from 'WAIT_REFRESH' procedure of AWARD BIOS - 1999
3976                            <1>     ;
3977                            <1> ;WAIT_REFRESH:  Uses port 61, bit 4 to have CPU speed independent waiting.
3978                            <1> ;   INPUT:  CX = number of refresh periods to wait
3979                            <1> ;                   (refresh periods = 1 per 30 microseconds on most machines)
3980                            <1> WR_STATE_0:
3981 00002424 E461              <1>     in      al, PORT_B              ; IN AL,SYS1
3982 00002426 A810              <1>     test    al, 010h
3983 00002428 74FA              <1>     JZ      short WR_STATE_0
3984                            <1> WR_STATE_1:
3985 0000242A E461              <1>     in      al, PORT_B              ; IN AL,SYS1
3986 0000242C A810              <1>     test    al, 010h
3987 0000242E 75FA              <1>     jnz     short WR_STATE_1
3988 00002430 E2F2              <1>         loop    WR_STATE_0
3989                            <1>     ;
3990                            <1>     ;pop    ax                      ; RESTORE (AH)
3991                            <1>     ; 11/04/2021
3992 00002432 58                <1>     pop     eax
3993 00002433 C3                <1>     retn                            ; (CX) = 0
3994                            <1>
3995                            <1> ; 03/08/2022
3996                            <1> ; 02/08/2022 - TRDOS 386 Kernel v2.0.5
3997                            <1> ; 09/07/2016
3998                            <1> ; 01/07/2016
3999                            <1> ; 24/06/2016
4000                            <1> ; 23/06/2016 - TRDOS 386 (TRDOS v2.0)
4001                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
4002                            <1> ;-----------------------------------------------------------------------------
4003                            <1> ; WRITE_STRING                                                          :
4004                            <1> ;   THIS ROUTINE WRITES A STRING OF CHARACTERS TO THE CRT.              :
4005                            <1> ; INPUT                                                         :
4006                            <1> ;   (AL) = WRITE STRING COMMAND  0 - 3                          :
4007                            <1> ;   (BH) = DISPLAY PAGE (ACTIVE PAGE)                           :
4008                            <1> ;   (CX) = COUNT OF CHARACTERS TO WRITE, IF (CX) = 0 THEN RETURN        :
4009                            <1> ;   (DX) = CURSOR POSITION FOR START OF STRING WRITE            :
4010                            <1> ;   (BL) = ATTRIBUTE OF CHARACTER TO WRITE IF (AL) = 0  OR     (AL) = 1       :
4011                            <1> ;   (EBP) = SOURCE STRING OFFSET                                :
4012                            <1> ; OUTPUT                                                        :
4013                            <1> ;   NONE                                                        :
4014                            <1> ;-----------------------------------------------------------------------------
4015                            <1>
4016                            <1> ; AL = 00h: Assign all characters the attribute in BL; do not update cursor
4017                            <1> ; AL = 01h: Assign all characters the attribute in BL; update cursor
```

```
                        <1> ; AL = 02h: Use attributes in string; do not update cursor
                        <1> ; AL = 03h: Use attributes in string; update cursor
                        <1>
                        <1> WRITE_STRING:
                        <1>     ; 03/08/2022
                        <1>     ; 02/08/2022
                        <1>     ; 12/09/2016
                        <1>     ; 09/07/2016
                        <1>     ;cmp    byte [CRT_MODE], 7 ; 6?!
                        <1>     ;ja     VIDEO_RETURN           ; not a valid function for VGA modes
                        <1>     ;
00002434 A2[14830100]   <1>     mov     [w_str_cmd], al               ; save (AL) command
00002439 3C04           <1>     cmp     al, 4                  ; TEST FOR INVALID WRITE STRING OPTION
                        <1>     ;jnb    VIDEO_RETURN           ; IF OPTION INVALID THEN RETURN
                        <1>     ; 02/08/2022
0000243B 7361           <1>     jnb     short P55
                        <1>
                        <1>         ;jcxz      VIDEO_RETURN           ; IF ZERO LENGTH STRING THEN RETURN
                        <1>
0000243D 67E35E         <1>         jcxz    P55                    ; 01/07/2016
                        <1>
                        <1>     ; 01/07/2016
                        <1>     ;and    ecx, 0FFFFh
                        <1>     ; ecx = byte count
                        <1>     ;push   ecx
00002440 89EE           <1>     mov     esi, ebp ; user buffer
00002442 BF00000700     <1>     mov     edi, Cluster_Buffer  ; system buffer
00002447 E8E8E60000     <1>     call    transfer_from_user_buffer
                        <1>     ;pop    ecx
                        <1>     ;jc     VIDEO_RETURN
                        <1>     ; 02/08/2022
0000244C 7250           <1>     jc      short P55
                        <1>     ; ecx = transfer (byte) count = character count
0000244E BD00000700     <1>     mov     ebp, Cluster_Buffer
                        <1>     ; 12/09/2016
00002453 803D[1E680000]07   <1>     cmp     byte [CRT_MODE], 7 ; 6?!
                        <1>     ;ja     vga_write_string
                        <1>     ; 02/08/2022
0000245A 7605           <1>     jna     short P57
0000245C E99C000000     <1>     jmp     vga_write_string
                        <1> P57:
00002461 0FB6F7         <1>     movzx   esi, bh                ; GET CURRENT CURSOR PAGE
                        <1>     ;sal    si, 1                  ; CONVERT TO PAGE OFFSET (SI = PAGE)
                        <1>     ; 02/08/2022
00002464 D1E6           <1>     sal     esi, 1
                        <1>     ; *****
00002466 66FFB6[9E760100]   <1>     push    word [esi+CURSOR_POSN]; SAVE CURRENT CURSOR POSITION IN STACK
                        <1>
                        <1>     ;mov    ax, 0200h              ; SET NEW CURSOR POSITION
                        <1>     ;int    10h
                        <1> P50next:
0000246D 51             <1>     push    ecx ; ****
0000246E 53             <1>     push    ebx ; *** ; 18/11/2020
0000246F 56             <1>     push    esi ; **
00002470 52             <1>     push    edx ; *
00002471 E8FDFEFFFF     <1>     call    _set_cpos
                        <1> P50:
00002476 8A4500         <1>     mov     al, [ebp]              ; GET CHARACTER FROM INPUT STRING
00002479 45             <1>     inc     ebp                    ; BUMP POINTER TO CHARACTER
                        <1>
                        <1> ;-----      TEST FOR SPECIAL CHARACTER'S
                        <1>
0000247A 3C08           <1>     cmp     al, 08h                ; IS IT A BACKSPACE
0000247C 7410           <1>     je      short P51              ; BACK_SPACE
0000247E 3C0D           <1>     cmp     al, 0Dh ; CR           ; IS IT CARRIAGE RETURN
00002480 740C           <1>     je      short P51              ; CAR_RET
00002482 3C0A           <1>     cmp     al, 0Ah ; LF           ; IS IT A LINE FEED
00002484 7408           <1>     je      short P51              ; LINE_FEED
                        <1>     ; 18/11/2020
00002486 3C09           <1>     cmp     al, 09h                ; is it a tab stop
00002488 7404           <1>     je      short P51
                        <1>     ;
0000248A 3C07           <1>     cmp     al, 07h                ; IS IT A BELL
0000248C 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>
0000248E E85BFEFFFF     <1>     call    _write_tty_m3
                        <1>
00002493 5A             <1>     pop     edx ; *
00002494 5E             <1>     pop     esi ; **
                        <1>
00002495 668B96[9E760100]   <1>     mov     dx, [esi+CURSOR_POSN] ; GET CURRENT CURSOR POSITION
0000249C EB44           <1>     jmp     short P54              ; SET CURSOR POSITION AND CONTINUE
                        <1> P55:
0000249E E9D5F6FFFF     <1>     jmp     VIDEO_RETURN
                        <1> P52:
                        <1>     ;mov    cx, 1                  ; SET CHARACTER WRITE AMOUNT TO ONE
                        <1>     ; 02/08/2022
000024A3 29C9           <1>     sub     ecx, ecx
000024A5 FEC1           <1>     inc     cl
                        <1>     ; ecx = 1
000024A7 803D[14830100]02   <1>     cmp     byte [w_str_cmd], 2    ; IS THE ATTRIBUTE IN THE STRING
000024AE 7204           <1>     jb      short P53              ; IF NOT THEN SKIP
000024B0 8A5D00         <1>     mov     bl, [ebp]              ; ELSE GET NEW ATTRIBUTE
000024B3 45             <1>     inc     ebp                    ; BUMP STRING POINTER
                        <1> P53:
                        <1>     ;mov    ah, 09h                ; GOT_CHARACTER
                        <1>     ;int    10h                    ; WRITE CHARACTER TO THE CRT
                        <1>
000024B4 E8A1FDFFFF     <1>     call    _write_c_current
                        <1>
000024B9 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>
000024BA 0FB6F7         <1>     movzx   esi, bh ; video page number (0 to 7)
000024BD 889E[27680000]     <1>     mov     [esi+chr_attrib], bl ; color/attribute
                        <1>
000024C3 FEC2           <1>     inc     dl                     ; INCREMENT COLUMN COUNTER
000024C5 3A15[20680000]     <1>     cmp     dl, [CRT_COLS]         ; IF COLS ARE WITHIN RANGE FOR THIS MODE
                        <1>     ;jb     short P54              ;    THEN GO TO COLUMNS SET
000024CB 7214           <1>     jb      short P56 ; 05/12/2020
000024CD FEC6           <1>     inc     dh                     ; BUMP ROW COUNTER BY ONE
000024CF 28D2           <1>     sub     dl, dl                 ; SET COLUMN COUNTER TO ZERO
000024D1 80FE19         <1>     cmp     dh, 25                 ; IF ROWS ARE LESS THAN 25 THEN
                        <1>     ;jb     short P54              ; GO TO ROWS_COLUMNS_SET
000024D4 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
000024D6 B00A               <1>     mov     al, 0Ah ; line feed
                            <1>
000024D8 E811FEFFFF         <1>     call    _write_tty_m3
                            <1>
000024DD 66BA0018           <1>     mov     dx, 1800h               ; Column = 0, Row = 24
                            <1> P56:
                            <1>     ; 05/12/2020
                            <1>     ; 18/11/2020
000024E1 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
000024E2 5B                 <1>     pop     ebx ; ***
000024E3 59                 <1>     pop     ecx ; ****
                            <1>
                            <1>     ;loop   P50                     ; DO IT ONCE MORE UNTIL (CX) = ZERO
000024E4 6649               <1>     dec     cx
000024E6 7585               <1>     jnz     short P50next
                            <1>
000024E8 665A               <1>     pop     dx  ; *****             ; RESTORE OLD CURSOR COORDINATES
                            <1>
000024EA F605[14830100]01   <1>     test    byte [w_str_cmd], 1     ; IF CURSOR WAS NOT TO BE MOVED
                            <1>     ;jnz    VIDEO_RETURN            ; THEN EXIT WITHOUT RESETTING OLD VALUE
                            <1>     ; 03/08/2022
000024F1 7505               <1>     jnz     short P58
                            <1>
                            <1>     ;mov    ax, 0200h               ; ELSE RESTORE OLD CURSOR POSITION
                            <1>     ;int    10h
                            <1>                                     ; DONE - EXIT WRITE STRING
000024F3 E87BFEFFFF         <1>     call    _set_cpos
                            <1> P58:
000024F8 E97BF6FFFF         <1>     jmp     VIDEO_RETURN            ; RETURN TO CALLER
                            <1>
                            <1> vga_write_string:
                            <1>     ; 04/08/2022 - TRDOS 386 Kernel v2.0.5
                            <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>
                            <1>     ; INPUT                                                         :
                            <1>     ;       (AL) = WRITE STRING COMMAND  0 - 3                      :
                            <1>     ;       (BH) = DISPLAY PAGE (ACTIVE PAGE)                       :
                            <1>     ;       (CX) = COUNT OF CHARACTERS TO WRITE, IF (CX) = 0 THEN RETURN     :
                            <1>     ;       (DX) = CURSOR POSITION FOR START OF STRING WRITE         :
                            <1>     ;       (BL) = ATTRIBUTE OF CHARACTER TO WRITE IF (AL) = 0  OR     (AL) = 1  :
                            <1>     ;       (EBP) = SOURCE STRING OFFSET                            :
                            <1>     ; OUTPUT                                                        :
                            <1>     ;       NONE                                                    :
                            <1>     ;-------------------------------------------------------------------------;
                            <1>
                            <1>     ; AL = 00h: Assign all characters the attribute in BL; do not update cursor
                            <1>     ; AL = 01h: Assign all characters the attribute in BL; update cursor
                            <1>     ; AL = 02h: Use attributes in string; do not update cursor
                            <1>     ; AL = 03h: Use attributes in string; update cursor
                            <1>
                            <1>     ; biosfn_write_string(GET_AL(),GET_BH(),GET_BL(),CX,GET_DH(),GET_DL(),ES,BP);
                            <1>     ; static void biosfn_write_string (flag,page,attr,count,row,col,seg,offset)
                            <1>
                            <1>     ; // Read curs info for the page
                            <1>     ; biosfn_get_cursor_pos(page,&dummy,&oldcurs);
                            <1>     ; bh = video page = 0
                            <1>     ;movzx  esi, word [CURSOR_POSN] ; current cursor position for video page 0
                            <1>
                            <1>     ; // if row=0xff special case : use current cursor position
                            <1>     ; if(row==0xff)
                            <1>     ;  {col=oldcurs&0x00ff;
                            <1>     ;   row=(oldcurs&0xff00)>>8;
                            <1>     ;  }
                            <1>
                            <1>     ;mov    al, [w_str_cmd]
                            <1>
000024FD 80FEFF             <1>     cmp     dh, 0FFh
00002500 7407               <1>     je      short vga_wstr_1 ; user current cursor position
                            <1> vga_wstr_0:
                            <1>     ; set cursor position
00002502 668915[9E760100]   <1>     mov     [CURSOR_POSN], dx ; save cursor pos for pg 0
                            <1> vga_wstr_1:
00002509 66FF35[9E760100]   <1>     push    word [CURSOR_POSN] ; *
                            <1>
                            <1>     ; ebp = string offset in system buffer (user buffer was copied to)
                            <1>
                            <1>     ; while(count--!=0)
                            <1>     ;  {
                            <1>     ;   car=read_byte(seg,offset++);
                            <1>     ;   if((flag&0x02)!=0)
                            <1>     ;    attr=read_byte(seg,offset++);
                            <1>     ;    biosfn_write_teletype(car,page,attr,WITH_ATTR);
                            <1>     ;  }
                            <1>
                            <1>     ;push   eax ; **
                            <1>     ;test   al, 2
00002510 F605[14830100]02   <1>     test    byte [w_str_cmd], 2
00002517 751D               <1>     jnz     short vga_wstr_3
00002519 881D[AF760100]     <1>     mov     [ccolor], bl
                            <1> vga_wstr_2:
0000251F 51                 <1>     push    ecx
00002520 8A4500             <1>     mov     al, [ebp]
00002523 E88F0A0000         <1>     call    vga_write_teletype
00002528 59                 <1>     pop     ecx
00002529 6649               <1>     dec     cx
0000252B 741E               <1>     jz      short vga_wstr_4
0000252D 45                 <1>     inc     ebp
0000252E 8A1D[AF760100]     <1>     mov     bl, [ccolor]
00002534 EBE9               <1>     jmp     short vga_wstr_2
                            <1> vga_wstr_3:
00002536 51                 <1>     push    ecx
00002537 8A4500             <1>     mov     al, [ebp]
0000253A 45                 <1>     inc     ebp
0000253B 8A5D00             <1>     mov     bl, [ebp]
0000253E E8740A0000         <1>     call    vga_write_teletype
00002543 59                 <1>     pop     ecx
00002544 6649               <1>     dec     cx
00002546 7403               <1>     jz      short vga_wstr_4
00002548 45                 <1>     inc     ebp
00002549 EBEB               <1>     jmp     short vga_wstr_3
```

```
                            <1> vga_wstr_4:
                            <1>     ; // Set back curs pos
                            <1>     ; if((flag&0x01)==0)
                            <1>     ;  biosfn_set_cursor_pos(page,oldcurs);
                            <1>     ; }
                            <1>     ;pop    eax ; **
0000254B 665A               <1>     pop     dx ; word [CURSOR_POSN] ; *
                            <1>     ;test   al, 1
0000254D F605[14830100]01   <1>     test    byte [w_str_cmd], 1
                            <1>     ;jnz    VIDEO_RETURN
                            <1>     ; 04/08/2022
00002554 7507               <1>     jnz     short vga_wstr_5
00002556 668915[9E760100]   <1>     mov     [CURSOR_POSN], dx
                            <1> vga_wstr_5:
0000255D E916F6FFFF         <1>     jmp     VIDEO_RETURN
                            <1>
                            <1> ; 03/08/2022 - TRDOS 386 Kernel v2.0.5
                            <1> ; 07/07/2016
                            <1> ; 27/06/2016 - TRDOS 386 (TRDOS v2.0)
                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
                            <1> ;--------------------------------------------------------
                            <1> ;  SCROLL UP
                            <1> ;   THIS ROUTINE SCROLLS UP THE INFORMATION ON THE CRT
                            <1> ; ENTRY ---
                            <1> ;  CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL
                            <1> ;  DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL
                            <1> ;   BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS
                            <1> ;  BH = FILL VALUE FOR BLANKED LINES
                            <1> ;  AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE FIELD)
                            <1> ;  DS = DATA SEGMENT
                            <1> ;  ES = REGEN SEGMENT
                            <1> ; EXIT --
                            <1> ;  NOTHING, THE SCREEN IS SCROLLED
                            <1> ;--------------------------------------------------------
                            <1>
                            <1>     ; cl = upper left column
                            <1>     ; ch = upper left row
                            <1>     ; dl = lower rigth column
                            <1>     ; dh = lower right row
                            <1>     ;
                            <1>     ; al = line count (AL=0 means blank entire fields)
                            <1>     ; bl = fill value for blanked lines
                            <1>     ; bh = unused
                            <1>
                            <1> GRAPHICS_UP:
                            <1>     ; 07/07/2016
                            <1>     ; AH = Current video mode, [CRT_MODE]
00002562 80FC07             <1>     cmp     ah, 7
00002565 7762               <1>     ja      short vga_graphics_up
                            <1>     ;je     n0
                            <1>
00002567 88C7               <1>     mov     bh, al                  ; save line count in BH
                            <1>     ;mov    ax, cx                  ; GET UPPER LEFT POSITION INTO AX REG
                            <1>     ; 03/08/2022
00002569 89C8               <1>     mov     eax, ecx
                            <1>
                            <1> ;-----      USE CHARACTER SUBROUTINE FOR POSITIONING
                            <1> ;-----      ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE
                            <1>
0000256B E8BA050000         <1>     call    GRAPH_POSN
                            <1>     ;movzx  edi, ax                 ; SAVE RESULT AS DESTINATION ADDRESS
                            <1>     ; 03/08/2022
00002570 89C7               <1>     mov     edi, eax
                            <1>
                            <1> ;-----      DETERMINE SIZE OF WINDOW
                            <1>
00002572 6629CA             <1>     sub     dx, cx
00002575 6681C20101         <1>         add     dx, 101h                ; ADJUST VALUES
0000257A C0E602             <1>     sal     dh, 2                   ; MULTIPLY ROWS BY 4 AT 8 VERT DOTS/CHAR
                            <1>                                     ; AND EVEN/ODD ROWS
                            <1> ;-----      DETERMINE CRT MODE
                            <1>
0000257D 803D[1E680000]06   <1>     cmp     byte [CRT_MODE], 6      ; TEST FOR MEDIUM RES
00002584 7304               <1>         jnc short _R7_              ; FIND_SOURCE
                            <1>
                            <1> ;-----      MEDIUM RES UP
00002586 D0E2               <1>     sal     dl, 1                   ; # COLUMNS * 2, SINCE 2 BYTES/CHAR
                            <1>     ;sal    di, 1                   ; OFFSET *2 SINCE 2 BYTES/CHAR
                            <1>     ; 03/08/2022
00002588 D1E7               <1>     sal     edi, 1
                            <1>
                            <1> ;-----      DETERMINE THE SOURCE ADDRESS IN THE BUFFER
                            <1> _R7_:                                   ; FIND_SOURCE
0000258A 81C700800B00       <1>     add     edi, 0B8000h
00002590 C0E702             <1>     sal     bh, 2                   ; multiply number of lines by 4
00002593 7430               <1>         jz  short _R11              ; IF ZERO, THEN BLANK ENTIRE FIELD
00002595 B050               <1>     mov     al, 80                  ; 80 BYTES/ROW
00002597 F6E7               <1>     mul     bh                      ; determine offset to source
                            <1>     ;movzx  esi, ax                 ; offset to source
                            <1>     ; 03/08/2022
00002599 89C6               <1>     mov     esi, eax
0000259B 01FE               <1>     add     esi, edi                ; SET UP SOURCE
0000259D 88F4               <1>     mov     ah, dh                  ; NUMBER OF ROWS IN FIELD
0000259F 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
000025A1 E8FD030000         <1>         call    _R17                    ; MOVE ONE ROW
000025A6 6681EEB01F         <1>     sub     si, 2000h-80            ; MOVE TO NEXT ROW
000025AB 6681EFB01F         <1>     sub     di, 2000h-80
000025B0 FECC               <1>     dec     ah                      ; NUMBER OF ROWS TO MOVE
000025B2 75ED               <1>         jnz     short _R8               ; CONTINUE TILL ALL MOVED
                            <1>
                            <1> ;-----      FILL IN THE VACATED LINE(S)
                            <1> _R9:                                    ; CLEAR ENTRY
000025B4 88D8               <1>     mov     al, bl                  ; attribute to fill with
                            <1> _R10_:
000025B6 E804040000         <1>         call    _R18                    ; CLEAR THAT ROW
000025BB 6681EFB01F         <1>     sub     di, 2000h-80            ; POINT TO NEXT LINE
000025C0 FECF               <1>     dec     bh                      ; number of lines to fill
000025C2 75F2               <1>     jnz     short _R10_             ; CLEAR LOOP
000025C4 C3                 <1>     retn                            ; EVERYYHING DONE
                            <1>
                            <1> _R11:                                   ; BLANK_FIELD
000025C5 88F7               <1>     mov     bh, dh                  ; set blank count to everything in field
000025C7 EBEB               <1>         jmp     short _R9               ; CLEAR THE FIELD
                            <1>
                            <1> vga_graphics_up:
                            <1>     ; 03/08/2022 - TRDOS 386 Kertnel v2.0.5
                            <1>     ; 12/04/2021
                            <1>     ; 08/08/2016
                            <1>     ; 07/08/2016
                            <1>     ; 04/08/2016
                            <1>     ; 01/08/2016
```

```
                                <1>     ; 31/07/2016
                                <1>     ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>     ;
                                <1>     ; derived from 'Plex86/Bochs VGABios' source code
                                <1>     ; vgabios-0.7a (2011)
                                <1>     ; by the LGPL VGABios developers Team (2001-2008)
                                <1>     ; 'vgabios.c', 'biosfn_scroll'
                                <1>     ;
                                <1>
                                <1>     ; cl = upper left column
                                <1>     ; ch = upper left row
                                <1>     ; dl = lower rigth column
                                <1>     ; dh = lower right row
                                <1>     ;
                                <1>     ; al = line count (AL=0 means blank entire fields)
                                <1>     ; bl = fill value for blanked lines
                                <1>     ; bh = unused
                                <1>     ;
                                <1>     ; ah = [CRT_MODE], current video mode
                                <1>
000025C9 88C7                  <1>     mov     bh, al ; 31/07/2016
000025CB BE[42680000]           <1>     mov     esi, vga_g_modes
000025D0 89F7                  <1>     mov     edi, esi
000025D2 83C708                <1>     add     edi, vga_g_mode_count
                                <1> vga_g_up_0:
000025D5 AC                    <1>     lodsb
000025D6 38E0                  <1>     cmp     al, ah ; [CRT_MODE]
000025D8 7405                  <1>     je      short vga_g_up_1
000025DA 39FE                  <1>     cmp     esi, edi
000025DC 72F7                  <1>     jb      short vga_g_up_0
                                <1>     ;xor    bh, bh ; 31/07/2016)
000025DE C3                    <1>     retn    ; nothing to do
                                <1> vga_g_up_1:
000025DF 88F8                  <1>     mov     al, bh ; 31/07/2016
000025E1 83C64F                <1>     add     esi, vga_g_memmodel - (vga_g_modes + 1)
                                <1>     ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
                                <1>
                                <1>     ; if(rlr>=nbrows)rlr=nbrows-1;
                                <1>     ; if(clr>=nbcols)clr=nbcols-1;
                                <1>     ; if(nblines>nbrows)nblines=0;
                                <1>     ; cols=clr-cul+1;
                                <1>
000025E4 3A35[26680000]         <1>     cmp     dh, [VGA_ROWS]
000025EA 7208                  <1>     jb      short vga_g_up_2
000025EC 8A35[26680000]         <1>     mov     dh, [VGA_ROWS]
000025F2 FECE                  <1>     dec     dh
                                <1> vga_g_up_2:
000025F4 3A15[20680000]         <1>     cmp     dl, [CRT_COLS]  ; = [VGA_COLS]
000025FA 7208                  <1>     jb      short vga_g_up_3
000025FC 8A15[20680000]         <1>     mov     dl, [CRT_COLS]
00002602 FECA                  <1>     dec     dl
                                <1> vga_g_up_3:
00002604 3A05[26680000]         <1>     cmp     al, [VGA_ROWS]
0000260A 7602                  <1>     jna     short vga_g_up_4
0000260C 28C0                  <1>     sub     al, al ; 0
                                <1> vga_g_up_4:
0000260E 88D7                  <1>     mov     bh, dl ; clr
00002610 28CF                  <1>     sub     bh, cl ; cul
00002612 FEC7                  <1>     inc     bh ; cols = clr-cul+1
                                <1>
00002614 20C0                  <1>     and     al, al ; nblines = 0
00002616 7559                  <1>     jnz     short vga_g_up_6
00002618 20ED                  <1>     and     ch, ch ; rul = 0
0000261A 7555                  <1>     jnz     short vga_g_up_6
0000261C 20C9                  <1>     and     cl, cl ; cul = 0
0000261E 7551                  <1>     jnz     short vga_g_up_6
                                <1>
                                <1>     ;push   ax
                                <1>     ; 12/04/2021
00002620 50                    <1>     push    eax
00002621 A0[26680000]           <1>     mov     al, [VGA_ROWS]
00002626 FEC8                  <1>     dec     al
00002628 38C6                  <1>     cmp     dh, al ; rlr = nbrows-1
0000262A 7545                  <1>     jne     short vga_g_up_5
0000262C A0[20680000]           <1>         mov     al, [CRT_COLS]  ; = VGA_COLS
00002631 FEC8                  <1>     dec     al
00002633 38C2                  <1>     cmp     dl, al ; clr = nbcols-1
                                <1>     ;jne    short vga_g_up_5
                                <1>     ;;pop   ax
                                <1>     ; 12/04/2021
00002635 58                    <1>     pop     eax
00002636 7539                  <1>     jne     short vga_g_up_5
                                <1>
00002638 66B80502              <1>     mov     ax, 0205h
0000263C 66BACE03              <1>     mov     dx, 3CEh ; VGAREG_GRDC_ADDRESS
00002640 66EF                  <1>     out     dx, ax
00002642 A0[26680000]           <1>     mov     al, [VGA_ROWS]
00002647 8A25[20680000]         <1>     mov     ah, [CRT_COLS] ; = [VGA_COLS]
0000264D F6E4                  <1>     mul     ah
0000264F 0FB7D0                <1>     movzx   edx, ax
                                <1>     ; 08/08/2016
00002652 0FB605[22680000]       <1>     movzx   eax, byte [CHAR_HEIGHT]
00002659 F7E2                  <1>     mul     edx
                                <1>     ; eax = byte count
0000265B 89C1                  <1>     mov     ecx, eax
                                <1>     ;; 07/08/2016
                                <1>     ;shl    dx, 3 ; * 8 ; * [CHAR_HEIGHT]
                                <1>     ;mov    ecx, edx
0000265D 88D8                  <1>     mov     al, bl ; fill value for blanked lines
0000265F BF00000A00            <1>     mov     edi, 0A0000h
00002664 F3AA                  <1>     rep     stosb
                                <1>
                                <1>     ;mov    ax, 5
                                <1>     ; 03/08/2022
00002666 30E4                  <1>     xor     ah, ah
00002668 B005                  <1>     mov     al, 5
                                <1>
0000266A 66BACE03              <1>     mov     dx, 3CEh ; VGAREG_GRDC_ADDRESS
0000266E 66EF                  <1>     out     dx, ax ; 0005h
                                <1>
00002670 C3                    <1>     retn
                                <1>
                                <1> vga_g_up_5:
                                <1>     ;;pop   ax
                                <1>     ; 12/04/2021
                                <1>     ;pop    eax
                                <1>
                                <1> vga_g_up_6:
                                <1>     ; [ESI] = VGA memory model number for current video mode
                                <1>         ;
                                <1>         ; LINEAR8 equ 5
                                <1>         ; PLANAR4 equ 4
                                <1>         ; PLANAR1 equ 3
                                <1>
```

```
4514 00002671 803E04              <1>     cmp     byte [esi], PLANAR4
4515 00002674 7424                <1>     je      short vga_g_up_planar
4516 00002676 803E03              <1>     cmp     byte [esi], PLANAR1
4517 00002679 741F                <1>     je      short vga_g_up_planar
4518                              <1> vga_g_up_linear8:
4519                              <1>     ; 07/07/2016 (TEMPORARY)
4520                              <1>     ;
4521                              <1>     ; cl = upper left column ; cul
4522                              <1>     ; ch = upper left row ; rul
4523                              <1>     ; dl = lower rigth column ; clr
4524                              <1>     ; dh = lower right row ; rlr
4525                              <1>
4526                              <1> vga_g_up_l0:
4527                              <1>     ;{for(i=rul;i<=rlr;i++)
4528                              <1>     ; if((i+nblines>rlr)||(nblines==0))
4529 0000267B 08C0                <1>     or      al, al
4530 0000267D 7414                <1>     jz      short vga_g_up_l2
4531 0000267F 88C4                <1>     mov     ah, al
4532 00002681 00EC                <1>     add     ah, ch ; i+nblines
4533                              <1>     ;jc     short vga_g_up_l2
4534 00002683 38F4                <1>     cmp     ah, dh
4535 00002685 770C                <1>     ja      short vga_g_up_l2
4536                              <1>     ; else
4537                              <1>     ;  vgamem_copy_pl4(cul,i+nblines,i,cols,nbcols,cheight);
4538 00002687 E8F2000000          <1>     call    vgamem_copy_l8
4539                              <1> vga_g_up_l1:
4540 0000268C FEC5                <1>     inc     ch
4541 0000268E 38F5                <1>     cmp     ch, dh
4542 00002690 76E9                <1>     jna     short vga_g_up_l0
4543 00002692 C3                  <1>     retn
4544                              <1> vga_g_up_l2:
4545                              <1>     ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
4546 00002693 E84C010000          <1>     call    vgamem_fill_l8
4547 00002698 EBF2                <1>     jmp     short vga_g_up_l1
4548                              <1>
4549                              <1> vga_g_up_planar:
4550                              <1>     ; cl = upper left column ; cul
4551                              <1>     ; ch = upper left row ; rul
4552                              <1>     ; dl = lower rigth column ; clr
4553                              <1>     ; dh = lower right row ; rlr
4554                              <1> vga_g_up_pl0:
4555                              <1>     ;{for(i=rul;i<=rlr;i++)
4556                              <1>     ; if((i+nblines>rlr)||(nblines==0))
4557 0000269A 20C0                <1>     and     al, al
4558 0000269C 7414                <1>     jz      short vga_g_up_pl2
4559 0000269E 88C4                <1>     mov     ah, al
4560 000026A0 00EC                <1>     add     ah, ch ; i+nblines
4561                              <1>     ;jc     short vga_g_up_pl2
4562 000026A2 38F4                <1>     cmp     ah, dh
4563 000026A4 770C                <1>     ja      short vga_g_up_pl2
4564                              <1>     ; else
4565                              <1>     ;  vgamem_copy_pl4(cul,i+nblines,i,cols,nbcols,cheight);
4566 000026A6 E80E000000          <1>     call    vgamem_copy_pl4
4567                              <1> vga_g_up_pl1:
4568 000026AB FEC5                <1>     inc     ch
4569 000026AD 38F5                <1>     cmp     ch, dh
4570 000026AF 76E9                <1>     jna     short vga_g_up_pl0
4571 000026B1 C3                  <1>     retn
4572                              <1> vga_g_up_pl2:
4573                              <1>     ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
4574 000026B2 E870000000          <1>     call    vgamem_fill_pl4
4575 000026B7 EBF2                <1>     jmp     short vga_g_up_pl1
4576                              <1>
4577                              <1> vgamem_copy_pl4:
4578                              <1>     ; 08/08/2016
4579                              <1>     ; 07/08/2016
4580                              <1>     ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
4581                              <1>     ;
4582                              <1>     ; derived from 'Plex86/Bochs VGABios' source code
4583                              <1>     ; vgabios-0.7a (2011)
4584                              <1>     ; by the LGPL VGABios developers Team (2001-2008)
4585                              <1>     ; 'vgabios.c', 'vgamem_copy_pl4'
4586                              <1>     ;
4587                              <1>     ; vgamem_copy_pl4(xstart,ysrc,ydest,cols,nbcols,cheight)
4588                              <1>     ; cl = xstart, ah = ysrc (i+nblines), ch = ydest (i),
4589                              <1>     ; bh = cols, [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
4590                              <1>
4591                              <1>     ; src=ysrc*cheight*nbcols+xstart;
4592                              <1>     ; dest=ydest*cheight*nbcols+xstart;
4593                              <1>
4594 000026B9 52                  <1>     push    edx
4595 000026BA 50                  <1>     push    eax
4596                              <1>
4597                              <1>     ; outw(VGAREG_GRDC_ADDRESS, 0x0105)
4598 000026BB 66B80501            <1>     mov     ax, 0105h
4599 000026BF 66BACE03            <1>     mov     dx, 3CEh ; VGAREG_GRDC_ADDRESS
4600 000026C3 66EF                <1>     out     dx, ax
4601                              <1>
4602                              <1>     ; 07/08/2016
4603                              <1>     ;mov     ah, [esp+1]
4604                              <1>     ;movzx  edx, ah ; ysrc
4605 000026C5 0FB6542401          <1>     movzx   edx, byte [esp+1]
4606                              <1>     ; 08/08/2016
4607 000026CA 0FB605[22680000]    <1>     movzx   eax, byte [CHAR_HEIGHT]
4608 000026D1 8A25[20680000]      <1>     mov     ah, [CRT_COLS] ; nbcols
4609 000026D7 F6E4                <1>     mul     ah
4610                              <1>     ;; 07/08/2016
4611                              <1>     ;movzx  eax, byte [CRT_COLS]
4612                              <1>     ;shl    ax, 3 ; * 8 ; * [CHAR_HEIGHT]
4613 000026D9 50                  <1>     push    eax ; cheight * nbcols
4614 000026DA F7E2                <1>     mul     edx ; * ysrc
4615                              <1>     ; eax = ysrc * cheight * nbcols
4616                              <1>     ; edx = 0
4617 000026DC 88CA                <1>     mov     dl, cl ; edx = xstart
4618 000026DE 01D0                <1>     add     eax, edx
4619 000026E0 89C6                <1>     mov     esi, eax ; src
4620 000026E2 88EA                <1>     mov     dl, ch ; ydest
4621 000026E4 58                  <1>     pop     eax ; cheight * nbcols
4622 000026E5 F7E2                <1>     mul     edx
4623                              <1>     ; eax = ydest * cheight * nbcols
4624 000026E7 88CA                <1>     mov     dl, cl ; edx = xstart
4625 000026E9 01D0                <1>     add     eax, edx
4626 000026EB 89C7                <1>     mov     edi, eax ; dest
4627                              <1>     ; esi = src
4628                              <1>     ; edi = dest
4629                              <1>     ; for(i=0;i<cheight;i++)
4630                              <1>     ; {
4631                              <1>     ;  memcpyb(0xa000,dest+i*nbcols,0xa000,src+i*nbcols,cols);
4632                              <1>     ; }
4633 000026ED 51                  <1>     push    ecx
4634 000026EE B900000A00          <1>     mov     ecx, 0A0000h
4635 000026F3 01CE                <1>     add     esi, ecx
4636 000026F5 01CF                <1>     add     edi, ecx
4637                              <1>     ; 08/08/2016
```

```
4638 000026F7 8A35[22680000]      <1>     mov     dh, [CHAR_HEIGHT]
4639                              <1>     ;; 07/08/2016
4640                              <1>     ;mov    dh, 8 ; 07/08/2016
4641 000026FD 28D2                <1>     sub     dl, dl ; i
4642                              <1> vgamem_copy_pl4_0:
4643 000026FF 56                  <1>     push    esi
4644 00002700 57                  <1>     push    edi
4645 00002701 0FB605[20680000]    <1>     movzx   eax, byte [CRT_COLS]
4646 00002708 F6E2                <1>     mul     dl
4647                              <1>     ; eax = i * nbcols
4648 0000270A 01C7                <1>     add     edi, eax ; dest+i*nbcols
4649 0000270C 01C6                <1>     add     esi, eax
4650 0000270E 0FB6CF              <1>     movzx   ecx, bh ; cols
4651 00002711 F3A4                <1>     rep     movsb
4652 00002713 5F                  <1>     pop     edi
4653 00002714 5E                  <1>     pop     esi
4654 00002715 FECE                <1>     dec     dh
4655 00002717 75E6                <1>     jnz     short vgamem_copy_pl4_0
4656                              <1> vgamem_copy_pl4_1:
4657 00002719 59                  <1>     pop     ecx
4658                              <1>
4659                              <1>     ; outw(VGAREG_GRDC_ADDRESS, 0x0005);
4660 0000271A 66B80500            <1>     mov     ax, 0005h
4661 0000271E 66BACE03            <1>     mov     dx, 3CEh ; VGAREG_GRDC_ADDRESS
4662 00002722 66EF                <1>     out     dx, ax
4663                              <1>
4664 00002724 58                  <1>     pop     eax
4665 00002725 5A                  <1>     pop     edx
4666                              <1>
4667 00002726 C3                  <1>     retn
4668                              <1>
4669                              <1> vgamem_fill_pl4:
4670                              <1>     ; 08/08/2016
4671                              <1>     ; 07/08/2016
4672                              <1>     ; 04/08/2016
4673                              <1>     ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
4674                              <1>     ;
4675                              <1>     ; derived from 'Plex86/Bochs VGABios' source code
4676                              <1>     ; vgabios-0.7a (2011)
4677                              <1>     ; by the LGPL VGABios developers Team (2001-2008)
4678                              <1>     ; 'vgabios.c', 'vgamem_fill_pl4'
4679                              <1>     ;
4680                              <1>     ; vgamem_fill_pl4(xstart,ystart,cols,nbcols,cheight,attr)
4681                              <1>     ; cl = xstart, edi = ch = ystart, bh = cols,
4682                              <1>     ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight, attr = 0
4683                              <1>
4684                              <1>     ; dest=ystart*cheight*nbcols+xstart;
4685 00002727 52                  <1>     push    edx
4686 00002728 50                  <1>     push    eax
4687                              <1>
4688                              <1>     ; outw(VGAREG_GRDC_ADDRESS, 0x0205)
4689 00002729 66B80502            <1>     mov     ax, 0205h
4690 0000272D 66BACE03            <1>     mov     dx, 3CEh ; VGAREG_GRDC_ADDRESS
4691 00002731 66EF                <1>     out     dx, ax
4692                              <1>
4693                              <1>             ; 08/08/2016
4694 00002733 0FB605[22680000]    <1>     movzx   eax, byte [CHAR_HEIGHT]
4695 0000273A F6E5                <1>     mul     ch
4696                              <1>     ;; 07/08/2016
4697                              <1>     ;movzx  eax, ch
4698                              <1>     ;shl    ax, 3 ; * 8 ; * [CHAR_HEIGHT]
4699 0000273C 0FB615[20680000]    <1>     movzx   edx, byte [CRT_COLS] ; = [VGA_COLS]
4700 00002743 F7E2                <1>     mul     edx
4701                              <1>     ; edx  = 0
4702 00002745 88CA                <1>     mov     dl, cl
4703 00002747 01D0                <1>     add     eax, edx
4704 00002749 89C7                <1>     mov     edi, eax
4705                              <1>     ; edi = dest
4706                              <1>     ; for(i=0;i<cheight;i++)
4707                              <1>     ; {
4708                              <1>     ;  memsetb(0xa000,dest+i*nbcols,attr,cols);
4709                              <1>     ; }
4710 0000274B 81C700000A00        <1>     add     edi, 0A0000h
4711 00002751 51                  <1>     push    ecx
4712                              <1>     ; 08/08/2016
4713 00002752 8A35[22680000]      <1>     mov     dh, [CHAR_HEIGHT]
4714                              <1>     ;; 07/08/2016
4715                              <1>     ;mov    dh, 8 ; 07/08/2016
4716 00002758 28D2                <1>     sub     dl, dl ; i
4717                              <1> vgamem_fill_pl4_0:
4718 0000275A 57                  <1>     push    edi
4719 0000275B 0FB605[20680000]    <1>     movzx   eax, byte [CRT_COLS]
4720 00002762 F6E2                <1>     mul     dl
4721                              <1>     ; eax = i * nbcols
4722 00002764 01C7                <1>     add     edi, eax ; dest+i*nbcols
4723 00002766 88D8                <1>     mov     al, bl ; attr ; 04/08/2016
4724 00002768 0FB6CF              <1>     movzx   ecx, bh ; cols
4725 0000276B F3AA                <1>     rep     stosb
4726 0000276D 5F                  <1>     pop     edi
4727 0000276E 75EA                <1>     jnz     short vgamem_fill_pl4_0
4728                              <1> vgamem_fill_pl4_1:
4729 00002770 59                  <1>     pop     ecx
4730                              <1>
4731                              <1>     ; outw(VGAREG_GRDC_ADDRESS, 0x0005);
4732 00002771 66B80500            <1>     mov     ax, 0005h
4733 00002775 66BACE03            <1>     mov     dx, 3CEh ; VGAREG_GRDC_ADDRESS
4734 00002779 66EF                <1>     out     dx, ax
4735                              <1>
4736 0000277B 58                  <1>     pop     eax
4737 0000277C 5A                  <1>     pop     edx
4738                              <1>
4739 0000277D C3                  <1>     retn
4740                              <1>
4741                              <1> vgamem_copy_l8:
4742                              <1>     ; 02/08/2022 - TRDOS 386 Kernel v2.0.5
4743                              <1>     ; 08/08/2016
4744                              <1>     ; 07/08/2016
4745                              <1>     ; 06/08/2016
4746                              <1>     ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
4747                              <1>     ;
4748                              <1>     ; TEMPORARY
4749                              <1>     ;
4750                              <1>     ; derived from 'Plex86/Bochs VGABios' source code
4751                              <1>     ; vgabios-0.7a (2011)
4752                              <1>     ; by the LGPL VGABios developers Team (2001-2008)
4753                              <1>     ; 'vgabios.c', 'vgamem_copy_pl4'
4754                              <1>     ;
4755                              <1>     ; vgamem_copy_pl4(xstart,ysrc,ydest,cols,nbcols,cheight)
4756                              <1>     ; cl = xstart, ah = ysrc (i+nblines), ch = ydest (i),
4757                              <1>     ; bh = cols, [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
4758                              <1>
4759                              <1>     ; src=ysrc*cheight*nbcols+xstart;
4760                              <1>     ; dest=ydest*cheight*nbcols+xstart;
4761                              <1>
```

```
4762 0000277E 52                  <1>     push    edx
4763 0000277F 50                  <1>     push    eax
4764                              <1>
4765                              <1>     ; outw(VGAREG_GRDC_ADDRESS, 0x0105)
4766                              <1>     ;mov    ax, 0105h
4767                              <1>     ;mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
4768                              <1>     ;out    dx, ax
4769                              <1>
4770                              <1>     ;mov    ah, [esp+1]
4771                              <1>
4772 00002780 0FB6D4              <1>     movzx   edx, ah ; ysrc
4773                              <1>     ; 08/08/2016
4774 00002783 0FB605[22680000]    <1>     movzx   eax, byte [CHAR_HEIGHT]
4775 0000278A 8A25[20680000]      <1>     mov     ah, [CRT_COLS] ; nbcols
4776 00002790 F6E4                <1>     mul     ah
4777                              <1>     ;; 07/08/2016
4778                              <1>     ;movzx  eax, byte [CRT_COLS]
4779                              <1>     ;shl    ax, 3 ; * 8 ; * [CHAR_HEIGHT]
4780 00002792 50                  <1>     push    eax ; cheight * nbcols
4781 00002793 F7E2                <1>     mul     edx ; * ysrc
4782                              <1>     ; eax = ysrc * cheight * nbcols
4783                              <1>     ; edx = 0
4784 00002795 88CA                <1>     mov     dl, cl ; edx = xstart
4785 00002797 01D0                <1>     add     eax, edx
4786 00002799 89C6                <1>     mov     esi, eax ; src
4787                              <1>     ;shl    si, 3 ; * 8 ; 06/08/2016
4788                              <1>     ; 02/08/2022
4789 0000279B C1E603              <1>     shl     esi, 3
4790 0000279E 88EA                <1>     mov     dl, ch ; ydest
4791 000027A0 58                  <1>     pop     eax ; cheight * nbcols
4792 000027A1 F7E2                <1>     mul     edx
4793                              <1>     ; eax = ydest * cheight * nbcols
4794 000027A3 88CA                <1>     mov     dl, cl ; edx = xstart
4795 000027A5 01D0                <1>     add     eax, edx
4796 000027A7 89C7                <1>     mov     edi, eax ; dest
4797                              <1>     ;shl    di, 3 ; * 8 ; 06/08/2016
4798                              <1>     ; 02/08/2022
4799 000027A9 C1E703              <1>     shl     edi, 3
4800                              <1>     ; esi = src
4801                              <1>     ; edi = dest
4802                              <1>     ; for(i=0;i<cheight;i++)
4803                              <1>     ; {
4804                              <1>     ;  memcpyb(0xa000,dest+i*nbcols,0xa000,src+i*nbcols,cols);
4805                              <1>     ; }
4806 000027AC 51                  <1>     push    ecx
4807 000027AD B900000A00          <1>     mov     ecx, 0A0000h
4808 000027B2 01CE                <1>     add     esi, ecx
4809 000027B4 01CF                <1>     add     edi, ecx
4810                              <1>     ; 08/08/2016
4811 000027B6 8A35[22680000]      <1>     mov     dh, [CHAR_HEIGHT]
4812                              <1>     ;; 07/08/2016
4813                              <1>     ;mov    dh, 8 ; 07/08/2016
4814 000027BC 28D2                <1>     sub     dl, dl ; i
4815                              <1> vgamem_copy_l8_0:
4816 000027BE 56                  <1>     push    esi
4817 000027BF 57                  <1>     push    edi
4818 000027C0 0FB605[20680000]    <1>     movzx   eax, byte [CRT_COLS]
4819 000027C7 F6E2                <1>     mul     dl
4820                              <1>     ; eax = i * nbcols
4821                              <1>     ;shl    ax, 3 ; * 8 ; 06/08/2016
4822                              <1>     ; 02/08/2022
4823 000027C9 C1E003              <1>     shl     eax, 3
4824 000027CC 01C7                <1>     add     edi, eax ; dest+i*nbcols
4825 000027CE 01C6                <1>     add     esi, eax
4826 000027D0 0FB6CF              <1>     movzx   ecx, bh ; cols
4827                              <1>     ;shl    cx, 3 ; * 8 ; 06/08/2016
4828                              <1>     ; 02/08/2022
4829 000027D3 C1E103              <1>     shl     ecx, 3
4830 000027D6 F3A4                <1>     rep     movsb
4831 000027D8 5F                  <1>     pop     edi
4832 000027D9 5E                  <1>     pop     esi
4833 000027DA FEC2                <1>     inc     dl ; 06/08/2016
4834 000027DC FECE                <1>     dec     dh
4835 000027DE 75DE                <1>     jnz     short vgamem_copy_l8_0
4836                              <1> vgamem_copy_l8_1:
4837 000027E0 59                  <1>     pop     ecx
4838                              <1>
4839                              <1>     ;; outw(VGAREG_GRDC_ADDRESS, 0x0005);
4840                              <1>     ;mov    ax, 0005h
4841                              <1>     ;mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
4842                              <1>     ;out    dx, ax
4843                              <1>
4844 000027E1 58                  <1>     pop     eax
4845 000027E2 5A                  <1>     pop     edx
4846                              <1>
4847 000027E3 C3                  <1>     retn
4848                              <1>
4849                              <1> vgamem_fill_l8:
4850                              <1>     ; 02/08/2022 - TRDOS 386 Kernel v2.0.5
4851                              <1>     ; 08/08/2016
4852                              <1>     ; 07/08/2016
4853                              <1>     ; 06/08/2016
4854                              <1>     ; 04/08/2016
4855                              <1>     ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
4856                              <1>     ;
4857                              <1>     ; TEMPORARY
4858                              <1>     ;
4859                              <1>     ; derived from 'Plex86/Bochs VGABios' source code
4860                              <1>     ; vgabios-0.7a (2011)
4861                              <1>     ; by the LGPL VGABios developers Team (2001-2008)
4862                              <1>     ; 'vgabios.c', 'vgamem_fill_pl4'
4863                              <1>     ;
4864                              <1>     ; vgamem_fill_pl4(xstart,ystart,cols,nbcols,cheight,attr)
4865                              <1>     ; cl = xstart, edi = ch = ystart, bh = cols,
4866                              <1>     ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight, attr = 0
4867                              <1>
4868                              <1>     ; dest=ystart*cheight*nbcols+xstart;
4869 000027E4 52                  <1>     push    edx
4870 000027E5 50                  <1>     push    eax
4871                              <1>
4872                              <1>     ;; outw(VGAREG_GRDC_ADDRESS, 0x0205)
4873                              <1>     ;mov    ax, 0205h
4874                              <1>     ;mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
4875                              <1>     ;out    dx, ax
4876                              <1>
4877                              <1>         ; 08/08/2016
4878 000027E6 0FB605[22680000]    <1>     movzx   eax, byte [CHAR_HEIGHT]
4879 000027ED F6E5                <1>     mul     ch
4880                              <1>     ;; 07/08/2016
4881                              <1>     ;movzx  eax, ch
4882                              <1>     ;shl    ax, 3 ; * 8 ; * [CHAR_HEIGHT]
4883 000027EF 0FB615[20680000]    <1>     movzx   edx, byte [CRT_COLS] ; = [VGA_COLS]
4884 000027F6 F7E2                <1>     mul     edx
4885                              <1>     ; edx  = 0
```

```
4886 000027F8 88CA              <1>     mov     dl, cl
4887 000027FA 01D0              <1>     add     eax, edx
4888 000027FC 89C7              <1>     mov     edi, eax
4889                            <1>     ;shl    di, 3 ; * 8 ; 06/08/2016
4890                            <1>     ; 02/08/2022
4891 000027FE C1E703            <1>     shl     edi, 3
4892                            <1>     ; edi = dest
4893                            <1>     ; for(i=0;i<cheight;i++)
4894                            <1>     ; {
4895                            <1>     ;  memsetb(0xa000,dest+i*nbcols,attr,cols);
4896                            <1>     ; }
4897 00002801 81C700000A00        <1>     add     edi, 0A0000h
4898 00002807 51                <1>     push    ecx
4899                            <1>     ; 08/08/2016
4900 00002808 8A35[22680000]    <1>     mov     dh, [CHAR_HEIGHT]
4901                            <1>     ;; 07/08/2016
4902                            <1>     ;mov    dh, 8 ; 07/08/2016
4903 0000280E 28D2              <1>     sub     dl, dl ; i
4904                            <1> vgamem_fill_18_0:
4905 00002810 57                <1>     push    edi
4906 00002811 0FB605[20680000]  <1>     movzx   eax, byte [CRT_COLS]
4907 00002818 F6E2              <1>     mul     dl
4908                            <1>     ; eax = i * nbcols
4909                            <1>     ;shl    ax, 3 ; * 8 ; 06/08/2016
4910                            <1>     ; 02/08/2022
4911 0000281A C1E003            <1>     shl     eax, 3
4912 0000281D 01C7              <1>     add     edi, eax ; dest+i*nbcols
4913 0000281F 88D8              <1>     mov     al, bl ; attr ; 04/08/2016
4914 00002821 0FB6CF            <1>     movzx   ecx, bh ; cols
4915                            <1>     ;shl    cx, 3 ; * 8 ; 06/08/2016
4916                            <1>     ; 02/08/2022
4917 00002824 C1E103            <1>     shl     ecx, 3
4918 00002827 F3AA              <1>     rep     stosb
4919 00002829 5F                <1>     pop     edi
4920 0000282A FEC2              <1>     inc     dl ; 06/08/2016
4921 0000282C FECE              <1>     dec     dh
4922 0000282E 75E0              <1>     jnz     short vgamem_fill_18_0
4923                            <1> vgamem_fill_18_1:
4924 00002830 59                <1>     pop     ecx
4925                            <1>
4926                            <1>     ;; outw(VGAREG_GRDC_ADDRESS, 0x0005);
4927                            <1>     ;mov    ax, 0005h
4928                            <1>     ;mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
4929                            <1>     ;out    dx, ax
4930                            <1>
4931 00002831 58                <1>     pop     eax
4932 00002832 5A                <1>     pop     edx
4933                            <1>
4934 00002833 C3                <1>     retn
4935                            <1>
4936                            <1> ; 03/08/2022 - TRDOS 386 Kernel v2.0.5
4937                            <1> ; 07/07/2016
4938                            <1> ; 27/06/2016 - TRDOS 386 (TRDOS v2.0)
4939                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
4940                            <1> ;--------------------------------------------------------
4941                            <1> ; SCROLL DOWN
4942                            <1> ;  THIS ROUTINE SCROLLS DOWN THE INFORMATION ON THE CRT
4943                            <1> ; ENTRY --
4944                            <1> ;  CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL
4945                            <1> ;  DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL
4946                            <1> ;   BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS
4947                            <1> ;  BH = FILL VALUE FOR BLANKED LINES
4948                            <1> ;  AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE FIELD)
4949                            <1> ;  DS = DATA SEGMENT
4950                            <1> ;  ES = REGEN SEGMENT
4951                            <1> ; EXIT --
4952                            <1> ;  NOTHING, THE SCREEN IS SCROLLED
4953                            <1> ;--------------------------------------------------------
4954                            <1>
4955                            <1>     ; cl = upper left column
4956                            <1>     ; ch = upper left row
4957                            <1>     ; dl = lower rigth column
4958                            <1>     ; dh = lower right row
4959                            <1>     ;
4960                            <1>     ; al = line count (AL=0 means blank entire fields)
4961                            <1>     ; bl = fill value for blanked lines
4962                            <1>     ; bh = unused
4963                            <1>
4964                            <1> GRAPHICS_DOWN:
4965                            <1>     ; 07/07/2016
4966                            <1>     ; ah = Current video mode, [CRT_MODE]
4967                            <1>     ; std                           ; SET DIRECTION
4968 00002834 80FC07            <1>     cmp     ah, 7
4969 00002837 7769              <1>         ja  short vga_graphics_down ; 03/08/2022
4970                            <1>     ;je     _n0
4971                            <1>
4972 00002839 88C7              <1>     mov     bh, al                  ; save line count in BH
4973                            <1>     ;mov    ax, dx                  ; GET LOWER RIGHT POSITION INTO AX REG
4974                            <1>     ; 03/08/2022
4975 0000283B 89D0              <1>     mov     eax, edx
4976                            <1>
4977                            <1> ;-----      USE CHARACTER SUBROUTINE FOR POSITIONING
4978                            <1> ;-----      ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE
4979                            <1>
4980 0000283D E8E8020000        <1>     call    GRAPH_POSN
4981                            <1>     ;movzx  edi, ax                 ; SAVE RESULT AS DESTINATION ADDRESS
4982                            <1>     ; 03/08/2022
4983 00002842 89C7              <1>     mov     edi, eax
4984                            <1>
4985                            <1> ;-----      DETERMINE SIZE OF WINDOW
4986                            <1>
4987 00002844 6629CA            <1>     sub     dx, cx
4988 00002847 6681C20101        <1>         add     dx, 101h                ; ADJUST VALUES
4989 0000284C C0E602            <1>     sal     dh, 2                   ; MULTIPLY ROWS BY 4 AT 8 VERT DOTS/CHAR
4990                            <1>                                     ; AND EVEN/ODD ROWS
4991                            <1> ;-----      DETERMINE CRT MODE
4992                            <1>
4993 0000284F 803D[1E680000]06  <1>     cmp     byte [CRT_MODE], 6      ; TEST FOR MEDIUM RES
4994 00002856 7305              <1>     jnc     short _R12                ; FIND_SOURCE_DOWN
4995                            <1>
4996                            <1> ;-----      MEDIUM RES DOWN
4997 00002858 D0E2              <1>     sal     dl, 1                   ; # COLUMNS * 2, SINCE 2 BYTES/CHAR
4998                            <1>     ;sal    di, 1                   ; OFFSET *2 SINCE 2 BYTES/CHAR
4999                            <1>     ; 03/08/2022
5000 0000285A D1E7              <1>     sal     edi, 1
5001                            <1>     ;inc    di                      ; POINT TO LAST BYTE
5002                            <1>     ; 03/08/2022
5003 0000285C 47                <1>     inc     edi
5004                            <1>
5005                            <1> ;-----      DETERMINE THE SOURCE ADDRESS IN THE BUFFER
5006                            <1>
5007                            <1> _R12:                                       ; FIND_SOURCE_DOWN
5008 0000285D 81C700800B00      <1>     add     edi, 0B8000h
5009 00002863 6681C7F000        <1>     add     di, 240                 ; POINT TO LAST ROW OF PIXELS
```

```
5010 00002868 C0E702            <1>     sal     bh, 2                   ; multiply number of lines by 4
5011 0000286B 74(06)            <1>     jz      short 6                 ; IF ZERO, THEN BLANK ENTIRE FIELD
5012 0000286D B050              <1>     mov     al, 80                  ; 80 BYTES/ROW
5013 0000286F F6E7              <1>     mul     bh                      ; determine offset to source
5014 00002871 89FE              <1>     mov     esi, edi                ; SET UP SOURCE
5015                            <1>     ;sub    si, ax                  ; SUBTRACT THE OFFSET
5016                            <1>     ; 03/08/2022
5017 00002873 29C6              <1>     sub     esi, eax
5018 00002875 88F4              <1>     mov     ah, dh                  ; NUMBER OF ROWS IN FIELD
5019 00002877 28FC              <1>     sub     ah, bh                  ; determine number to move
5020                            <1>
5021                            <1> ;-----      LOOP THROUGH, MOVING ONE ROW AT A TIME, BOTH EVEN AND ODD FIELDS
5022                            <1>
5023                            <1> _R13:                                       ; ROW_LOOP_DOWN
5024 00002879 E825010000        <1>         call    _R17                        ; MOVE ONE ROW
5025 0000287E 6681EE5020        <1>     sub     si, 2000h+80            ; MOVE TO NEXT ROW
5026 00002883 6681EF5020        <1>     sub     di, 2000h+80
5027 00002888 FECC              <1>     dec     ah                      ; NUMBER OF ROWS TO MOVE
5028 0000288A 75ED              <1>     jnz     short _R13                ; CONTINUE TILL ALL MOVED
5029                            <1>
5030                            <1> ;-----      FILL IN THE VACATED LINE(S)
5031                            <1> _R14:                                       ; CLEAR_ENTRY_DOWN
5032 0000288C 88D8              <1>     mov     al, bl                  ; attribute to fill with
5033                            <1> _R15_:                                      ; CLEAR_LOOP_DOWN
5034 0000288E E82C010000        <1>     call    _R18                    ; CLEAR A ROW
5035 00002893 6681EF5020        <1>     sub     di, 2000h+80            ; POINT TO NEXT LINE
5036 00002898 FECF              <1>     dec     bh                      ; number of lines to fill
5037 0000289A 75F2              <1>         jnz short _R15_             ; CLEAR_LOOP_DOWN
5038                            <1>     ; 18/11/2020
5039 0000289C FC                <1>     cld                             ; RESET THE DIRECTION FLAG
5040                            <1>
5041 0000289D C3                <1>     retn                            ; EVERYTHING DONE
5042                            <1>
5043                            <1> _R16:                                       ; BLANK_FIELD_DOWN
5044 0000289E 88F7              <1>     mov     bh, dh                  ; set blank count to everything in field
5045 000028A0 EBEA              <1>         jmp     short _R14              ; CLEAR THE FIELD
5046                            <1>
5047                            <1> vga_graphics_down:
5048                            <1>     ; 03/08/2022 - TRDOS 386 Kertnel v2.0.5
5049                            <1>     ; 12/04/2021
5050                            <1>     ; 08/08/2016
5051                            <1>     ; 07/08/2016
5052                            <1>     ; 31/07/2016
5053                            <1>     ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
5054                            <1>     ;
5055                            <1>     ; derived from 'Plex86/Bochs VGABios' source code
5056                            <1>     ; vgabios-0.7a (2011)
5057                            <1>     ; by the LGPL VGABios developers Team (2001-2008)
5058                            <1>     ; 'vgabios.c', 'biosfn_scroll'
5059                            <1>     ;
5060                            <1>
5061                            <1>     ; cl = upper left column
5062                            <1>     ; ch = upper left row
5063                            <1>     ; dl = lower rigth column
5064                            <1>     ; dh = lower right row
5065                            <1>     ;
5066                            <1>     ; al = line count (AL=0 means blank entire fields)
5067                            <1>     ; bl = fill value for blanked lines
5068                            <1>     ; bh = unused
5069                            <1>     ;
5070                            <1>     ; ah = [CRT_MODE], current video mode
5071                            <1>
5072 000028A2 FC                <1>         cld     ; !!! Clear direction flag !!!
5073                            <1>
5074 000028A3 88C7              <1>     mov     bh, al ; 31/07/2016
5075                            <1>
5076 000028A5 BE[3A680000]      <1>     mov     esi, vga_modes
5077 000028AA 89F7              <1>     mov     edi, esi
5078 000028AC 83C710            <1>     add     edi, vga_mode_count
5079                            <1> vga_g_down_0:
5080 000028AF AC                <1>     lodsb
5081 000028B0 38E0              <1>     cmp     al, ah ; [CRT_MODE]
5082 000028B2 7405              <1>     je      short vga_g_down_1
5083 000028B4 39FE              <1>     cmp     esi, edi
5084 000028B6 72F7              <1>     jb      short vga_g_down_0
5085                            <1>     ; xor   bh, bh  ; 31/07/2016
5086 000028B8 C3                <1>     retn    ; nothing to do
5087                            <1> vga_g_down_1:
5088 000028B9 88F8              <1>     mov     al, bh ; 31/07/2016
5089 000028BB 83C64F            <1>     add     esi, vga_memmodel - (vga_modes + 1)
5090                            <1>     ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
5091                            <1>
5092                            <1>     ; if(rlr>=nbrows)rlr=nbrows-1;
5093                            <1>     ; if(clr>=nbcols)clr=nbcols-1;
5094                            <1>     ; if(nblines>nbrows)nblines=0;
5095                            <1>     ; cols=clr-cul+1;
5096                            <1>
5097 000028BE 3A35[26680000]      <1>     cmp     dh, [VGA_ROWS]
5098 000028C4 7208              <1>     jb      short vga_g_down_2
5099 000028C6 8A35[26680000]      <1>     mov     dh, [VGA_ROWS]
5100 000028CC FECE              <1>     dec     dh
5101                            <1> vga_g_down_2:
5102 000028CE 3A15[20680000]      <1>     cmp     dl, [CRT_COLS]  ; = [VGA_COLS]
5103 000028D4 7208              <1>     jb      short vga_g_down_3
5104 000028D6 8A15[20680000]      <1>     mov     dl, [CRT_COLS]
5105 000028DC FECA              <1>     dec     dl
5106                            <1> vga_g_down_3:
5107 000028DE 3A05[26680000]      <1>     cmp     al, [VGA_ROWS]
5108 000028E4 7602              <1>     jna     short vga_g_down_4
5109 000028E6 28C0              <1>     sub     al, al ; 0
5110                            <1> vga_g_down_4:
5111 000028E8 88F7              <1>     mov     bh, dh ; clr
5112 000028EA 28CF              <1>     sub     bh, cl ; cul
5113 000028EC FEC7              <1>     inc     bh ; cols = clr-cul+1
5114                            <1>
5115 000028EE 20C0              <1>     and     al, al ; nblines = 0
5116 000028F0 7559              <1>     jnz     short vga_g_down_6
5117 000028F2 20ED              <1>     and     ch, ch ; rul = 0
5118 000028F4 7555              <1>     jnz     short vga_g_down_6
5119 000028F6 20C9              <1>     and     cl, cl ; cul = 0
5120 000028F8 7551              <1>     jnz     short vga_g_down_6
5121                            <1>
5122 000028FA 50                <1>     push    eax ; push ax ; 12/04/2021
5123 000028FB A0[26680000]      <1>     mov     al, [VGA_ROWS]
5124 00002900 FEC8              <1>     dec     al
5125 00002902 38C6              <1>     cmp     dh, al ; rlr = nbrows-1
5126 00002904 7545              <1>     jne     short vga_g_down_5
5127 00002906 A0[20680000]      <1>         mov     al, [CRT_COLS]  ; = VGA_COLS
5128 0000290B FEC8              <1>     dec     al
5129 0000290D 38C2              <1>     cmp     dl, al ; clr = nbcols-1
5130                            <1>     ;jne    short vga_g_down_5
5131                            <1>     ; 12/04/2021
5132 0000290F 58                <1>     pop     eax ; pop ax
5133 00002910 7539              <1>     jne     short vga_g_down_5
```

```
                                <1>
00002912 66B80502               <1>     mov     ax, 0205h
00002916 66BACE03               <1>     mov     dx, 3CEh ; VGAREG_GRDC_ADDRESS
0000291A 66EF                   <1>     out     dx, ax
0000291C A0[26680000]           <1>     mov     al, [VGA_ROWS]
00002921 8A25[20680000]         <1>     mov     ah, [CRT_COLS] ; = [VGA_COLS]
00002927 F6E4                   <1>     mul     ah
00002929 0FB7D0                 <1>     movzx   edx, ax
                                <1>     ; 08/08/2016
0000292C 0FB605[22680000]       <1>     movzx   eax, byte [CHAR_HEIGHT]
00002933 F7E2                   <1>     mul     edx
                                <1>     ; eax = byte count
00002935 89C1                   <1>     mov     ecx, eax
                                <1>     ;; 07/08/2016
                                <1>     ;shl    dx, 3 ; * 8 ; * [CHAR_HEIGHT]
                                <1>     ;mov    ecx, edx
00002937 88D8                   <1>     mov     al, bl ; fill value for blanked lines
00002939 BF00000A00             <1>     mov     edi, 0A0000h
0000293E F3AA                   <1>     rep     stosb
                                <1>
                                <1>     ; 03/08/2022
00002940 30E4                   <1>     xor     ah, ah ; 0
                                <1>
00002942 B005                   <1>     mov     al, 5
00002944 66BACE03               <1>     mov     dx, 3CEh ; VGAREG_GRDC_ADDRESS
00002948 66EF                   <1>     out     dx, ax ; 0005h
                                <1>
0000294A C3                     <1>     retn
                                <1>
                                <1> vga_g_down_5:
                                <1>     ; 12/04/2021
                                <1>     ;pop    eax  ; pop ax
                                <1>
                                <1> vga_g_down_6:
                                <1>     ; [ESI] = VGA memory model number for current video mode
                                <1>         ;
                                <1>         ; LINEAR8 equ 5
                                <1>         ; PLANAR4 equ 4
                                <1>         ; PLANAR1 equ 3
                                <1>
0000294B 803E04                 <1>     cmp     byte [esi], PLANAR4
0000294E 742C                   <1>     je      short vga_g_down_planar
00002950 803E03                 <1>     cmp     byte [esi], PLANAR1
00002953 7427                   <1>     je      short vga_g_down_planar
                                <1> vga_g_down_linear8:
                                <1>     ; 07/07/2016 (TEMPORARY)
                                <1>     ;
                                <1>     ; cl = upper left column ; cul
                                <1>     ; ch = upper left row ; rul
                                <1>     ; dl = lower rigth column ; clr
                                <1>     ; dh = lower right row ; rlr
                                <1>
                                <1> vga_g_down_l0:
                                <1>     ;{for(i=rlr;i>=rul;i--)
                                <1>         ; if((i<rul+nblines)||(nblines==0))
00002955 08C0                   <1>     or      al, al
00002957 741C                   <1>     jz      short vga_g_down_l2
00002959 88C4                   <1>     mov     ah, al
0000295B 00EC                   <1>     add     ah, ch
                                <1>     ;jc     short vga_g_down_l2
0000295D 86F5                   <1>     xchg    ch, dh
0000295F 38E5                   <1>     cmp     ch, ah
00002961 7212                   <1>     jb      short vga_g_down_l2
00002963 88EC                   <1>     mov     ah, ch
00002965 28C4                   <1>     sub     ah, al ; ah = i - nblines
                                <1>     ; else
                                <1>     ; vgamem_copy_pl4(cul,i,i-nblines,cols,nbcols,cheight);
00002967 E812FEFFFF             <1>     call    vgamem_copy_l8
                                <1> vga_g_down_l1:
0000296C 86EE                   <1>     xchg    dh, ch
0000296E FECE                   <1>     dec     dh
00002970 38EE                   <1>     cmp     dh, ch
00002972 73E1                   <1>     jnb     short vga_g_down_l0
00002974 C3                     <1>     retn
                                <1>
                                <1> vga_g_down_l2:
                                <1>     ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
00002975 E86AFEFFFF             <1>     call    vgamem_fill_l8
0000297A EBF0                   <1>     jmp     short vga_g_down_l1
                                <1>
                                <1> vga_g_down_planar:
                                <1>     ; cl = upper left column ; cul
                                <1>     ; ch = upper left row ; rul
                                <1>     ; dl = lower rigth column ; clr
                                <1>     ; dh = lower right row ; rlr
                                <1> vga_g_down_p0:
                                <1>     ;{for(i=rlr;i>=rul;i--)
                                <1>         ; if((i<rul+nblines)||(nblines==0))
0000297C 08C0                   <1>     or      al, al
0000297E 741C                   <1>     jz      short vga_g_down_p2
00002980 88C4                   <1>     mov     ah, al
00002982 00EC                   <1>     add     ah, ch
                                <1>     ;jc     short vga_g_down_p2
00002984 86F5                   <1>     xchg    ch, dh
00002986 38E5                   <1>     cmp     ch, ah
00002988 7212                   <1>     jb      short vga_g_down_p2
0000298A 88EC                   <1>     mov     ah, ch
0000298C 28C4                   <1>     sub     ah, al ; ah = i - nblines
                                <1>     ; else
                                <1>     ; vgamem_copy_pl4(cul,i,i-nblines,cols,nbcols,cheight);
0000298E E826FDFFFF             <1>     call    vgamem_copy_pl4
                                <1> vga_g_down_p1:
00002993 86EE                   <1>     xchg    dh, ch
00002995 FECE                   <1>     dec     dh
00002997 38EE                   <1>     cmp     dh, ch
00002999 73E1                   <1>     jnb     short vga_g_down_p0
0000299B C3                     <1>     retn
                                <1>
                                <1> vga_g_down_p2:
                                <1>     ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
0000299C E886FDFFFF             <1>     call    vgamem_fill_pl4
000029A1 EBF0                   <1>     jmp     short vga_g_down_p1
                                <1>
                                <1>
                                <1>
                                <1> ; 27/06/2016 - TRDOS 386 (TRDOS v2.0)
                                <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
                                <1>
                                <1> ;-----      ROUTINE TO MOVE ONE ROW OF INFORMATION
                                <1>
                                <1> _R17:
000029A3 0FB6CA                 <1>     movzx   ecx, dl                 ; NUMBER OF BYTES IN THE ROW
000029A6 56                     <1>     push    esi
000029A7 57                     <1>     push    edi                     ; SAVE POINTERS
```

```
5258 000029A8 F3A4              <1>     rep     movsb                   ; MOVE THE EVEN FIELD
5259 000029AA 5F                <1>     pop     edi
5260 000029AB 5E                <1>     pop     esi
5261 000029AC 6681C60020        <1>     add     si, 2000h
5262 000029B1 6681C70020        <1>     add     di, 2000h               ; POINT TO THE ODD FIELD
5263 000029B6 56                <1>     push    esi
5264 000029B7 57                <1>     push    edi                     ; SAVE THE POINTERS
5265 000029B8 88D1              <1>     mov     cl, dl                  ; COUNT BACK
5266 000029BA F3A4              <1>     rep     movsb                   ; MOVE THE ODD FIELD
5267 000029BC 5F                <1>     pop     edi
5268 000029BD 5E                <1>     pop     esi                     ; POINTERS BACK
5269 000029BE C3                <1>     retn                            ; RETURN TO CALLER
5270                            <1>
5271                            <1> ;-----      CLEAR A SINGLE ROW
5272                            <1>
5273                            <1> _R18:
5274 000029BF 0FB6CA            <1>     movzx   ecx, dl                 ; NUMBER OF BYTES IN FIELD
5275 000029C2 57                <1>     push    edi                     ; SAVE POINTER
5276 000029C3 F3AA              <1>     rep     stosb                   ; STORE THE NEW VALUE
5277 000029C5 5F                <1>     pop     edi                     ; POINTER BACK
5278 000029C6 6681C70020        <1>     add     di, 2000h               ; POINT TO ODD FIELD
5279 000029CB 57                <1>     push    edi
5280 000029CC 88D1              <1>     mov     cl, dl
5281 000029CE F3AA              <1>     rep     stosb                   ; FILL THE ODD FIELD
5282 000029D0 5F                <1>     pop     edi
5283 000029D1 C3                <1>     retn                            ; RETURN TO CALLER
5284                            <1>
5285                            <1> ; 03/08/2022 - TRDOS 386 Kernel v2.0.5
5286                            <1> ; 04/07/2016
5287                            <1> ; 01/07/2016
5288                            <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
5289                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
5290                            <1> ;-------------------------------------------------
5291                            <1> ; GRAPHICS WRITE
5292                            <1> ;  THIS ROUTINE WRITES THE ASCII CHARACTER TO THE CURRENT
5293                            <1> ;  POSITION ON THE SCREEN.
5294                            <1> ; ENTRY --
5295                            <1> ;  AL = CHARACTER TO WRITE
5296                            <1> ;  BL = COLOR ATTRIBUTE TO BE USED FOR FOREGROUND COLOR
5297                            <1> ;   IF BIT 7 IS SET, THE CHAR IS XOR'D INTO THE REGEN BUFFER
5298                            <1> ;   (0 IS USED FOR THE BACKGROUND COLOR)
5299                            <1> ;  CX = NUMBER OF CHARS TO WRITE
5300                            <1> ;  DS = DATA SEGMENT
5301                            <1> ;  ES = REGEN SEGMENT
5302                            <1> ; EXIT --
5303                            <1> ;  NOTHING IS RETURNED
5304                            <1> ;
5305                            <1> ; GRAPHICS READ
5306                            <1> ;  THIS ROUTINE READS THE ASCII CHARACTER AT THE CURRENT CURSOR
5307                            <1> ;  POSITION ON THE SCREEN BY MATCHING THE DOTS ON THE SCREEN TO THE
5308                            <1> ;  CHARACTER GENERATOR CODE POINTS
5309                            <1> ; ENTRY --
5310                            <1> ;  NONE (0 IS ASSUMED AS THE BACKGROUND COLOR)
5311                            <1> ; EXIT --
5312                            <1> ;  AL = CHARACTER READ AT THAT POSITION (0 RETURNED IF NONE FOUND)
5313                            <1> ;
5314                            <1> ; FOR BOTH ROUTINES, THE IMAGES USED TO FORM CHARS ARE CONTAINED IN ROM
5315                            <1> ;  FOR THE 1ST 128 CHARS.  TO ACCESS CHARS IN THE SECOND HALF, THE USER
5316                            <1> ;  MUST INITIALIZE THE VECTOR AT INTERRUPT 1FH (LOCATION 0007CH) TO
5317                            <1> ;  POINT TO THE USER SUPPLIED TABLE OF GRAPHIC IMAGES (8X8 BOXES).
5318                            <1> ;  FAILURE TO DO SO WILL CAUSE IN STRANGE RESULTS
5319                            <1> ;-------------------------------------------------
5320                            <1>
5321                            <1> GRAPHICS_WRITE:
5322 000029D2 25FF000000        <1>     and     eax, 0FFh               ; ZERO TO HIGH OF CODE POINT
5323 000029D7 50                <1>     push    eax                     ; SAVE CODE POINT VALUE
5324                            <1>
5325                            <1> ;-----      DETERMINE POSITION IN REGEN BUFFER TO PUT CODE POINTS
5326                            <1>
5327 000029D8 E846010000        <1>     call    S26                     ; FIND LOCATION IN REGEN BUFFER
5328 000029DD 89C7              <1>     mov     edi, eax                ; REGEN POINTER IN DI
5329                            <1>
5330                            <1> ;-----      DETERMINE REGION TO GET CODE POINTS FROM
5331                            <1>
5332 000029DF 58                <1>     pop     eax                     ; RECOVER CODE POINT
5333                            <1>
5334 000029E0 BE[104D0100]      <1>     mov     esi, CRT_CHAR_GEN       ; OFFSET OF IMAGES
5335                            <1>
5336                            <1> ;-----      DETERMINE GRAPHICS MODE IN OPERATION
5337                            <1>                                     ; DETERMINE_MODE
5338                            <1>     ;sal    ax, 3                   ; MULTIPLY CODE POINT VALUE BY 8
5339                            <1>     ; 03/08/2022
5340 000029E5 C1E003            <1>     sal     eax, 3
5341 000029E8 01C6              <1>     add     esi, eax                ; SI HAS OFFSET OF DESIRED CODES
5342                            <1>
5343 000029EA 803D[1E680000]06  <1>     cmp     byte [CRT_MODE], 6
5344 000029F1 7231              <1>     jc      short S6                ; TEST FOR MEDIUM RESOLUTION MODE
5345                            <1>
5346                            <1> ;-----      HIGH RESOLUTION MODE
5347                            <1>
5348 000029F3 81C700800B00      <1>     add     edi, 0B8000h
5349                            <1> S1:                                 ; HIGH_CHAR
5350 000029F9 57                <1>     push    edi                     ; SAVE REGEN POINTER
5351 000029FA 56                <1>     push    esi                     ; SAVE CODE POINTER
5352 000029FB B604              <1>     mov     dh, 4                   ; NUMBER OF TIMES THROUGH LOOP
5353                            <1> S2:
5354 000029FD AC                <1>     lodsb                           ; GET BYTE FROM CODE POINTS
5355 000029FE F6C380            <1>     test    bl, 80h                 ; SHOULD WE USE THE FUNCTION
5356 00002A01 7515              <1>     jnz     short S5                ; TO PUT CHAR IN
5357 00002A03 AA                <1>     stosb                           ; STORE IN REGEN BUFFER
5358 00002A04 AC                <1>     lodsb
5359                            <1> S4:
5360 00002A05 8887FF1F0000      <1>     mov     [edi+2000h-1], al       ; STORE IN SECOND HALF
5361 00002A0B 83C74F            <1>     add     edi, 79                 ; MOVE TO NEXT ROW IN REGEN
5362 00002A0E FECE              <1>     dec     dh                      ; DONE WITH LOOP
5363 00002A10 75EB              <1>     jnz     short S2
5364 00002A12 5E                <1>     pop     esi
5365 00002A13 5F                <1>     pop     edi                     ; RECOVER REGEN POINTER
5366 00002A14 47                <1>     inc     edi                     ; POINT TO NEXT CHAR POSITION
5367 00002A15 E2E2              <1>     loop    S1                      ; MORE CHARS TO WRITE
5368 00002A17 C3                <1>     retn
5369                            <1>
5370                            <1> S5:
5371 00002A18 3207              <1>     xor     al, [edi]               ; EXCLUSIVE OR WITH CURRENT
5372 00002A1A AA                <1>     stosb                           ; STORE THE CODE POINT
5373 00002A1B AC                <1>     lodsb                           ; AGAIN FOR ODD FIELD
5374 00002A1C 3287FF1F0000      <1>     xor     al, [edi+2000h-1]
5375 00002A22 EBE1              <1>     jmp     short S4                ; BACK TO MAINSTREAM
5376                            <1>
5377                            <1> ;-----      MEDIUM RESOLUTION WRITE
5378                            <1> S6:                                 ; MED_RES_WRITE
5379 00002A24 88DA              <1>     mov     dl, bl                  ; SAVE HIGH COLOR BIT
5380                            <1>     ; 03/08/2022
5381 00002A26 D1E7              <1>     sal     edi, 1
```

```
5382                                <1>     ;sal    di, 1                   ; OFFSET*2 SINCE 2 BYTES/CHAR
5383                                <1>                                     ; EXPAND BL TO FULL WORD OF COLOR
5384 00002A28 80E303               <1>     and     bl, 3                   ; ISOLATE THE COLOR BITS ( LOW 2 BITS )
5385 00002A2B B055                 <1>     mov     al, 055h                ; GET BIT CONVERSION MULTIPLIER
5386 00002A2D F6E3                 <1>     mul     bl                      ; EXPAND 2 COLOR BITS TO 4 REPLICATIONS
5387 00002A2F 88C3                 <1>     mov     bl, al                  ; PLACE BACK IN WORK REGISTER
5388 00002A31 88C7                 <1>     mov     bh, al                  ; EXPAND TO 8 REPLICATIONS OF COLOR BITS
5389 00002A33 81C700800B00         <1>     add     edi, 0B8000h
5390                                <1> S7:                                         ; MED_CHAR
5391 00002A39 57                   <1>     push    edi                     ; SAVE REGEN POINTER
5392 00002A3A 56                   <1>     push    esi                     ; SAVE THE CODE POINTER
5393 00002A3B B604                 <1>     mov     dh, 4                   ; NUMBER OF LOOPS
5394                                <1> S8:
5395 00002A3D AC                   <1>     lodsb                           ; GET CODE POINT
5396 00002A3E E8B1000000           <1>     call    S21                     ; DOUBLE UP ALL THE BITS
5397 00002A43 6621D8               <1>     and     ax, bx                  ; CONVERT TO FOREGROUND COLOR ( 0 BACK )
5398 00002A46 86C4                 <1>     xchg    ah, al                  ; SWAP HIGH/LOW BYTES FOR WORD MOVE
5399 00002A48 F6C280               <1>     test    dl, 80h                 ; IS THIS XOR FUNCTION
5400 00002A4B 7403                 <1>     jz      short S9                ; NO, STORE IT IN AS IS
5401 00002A4D 663307               <1>     xor     ax, [edi]               ; DO FUNCTION WITH LOW/HIGH
5402                                <1> S9:
5403 00002A50 668907               <1>     mov     [edi], ax               ; STORE FIRST BYTE HIGH, SECOND LOW
5404 00002A53 AC                   <1>     lodsb                           ; GET CODE POINT
5405 00002A54 E89B000000           <1>     call    S21
5406 00002A59 6621D8               <1>     and     ax, bx                  ; CONVERT TO COLOR
5407 00002A5C 86C4                 <1>     xchg    ah, al                  ; SWAP HIGH/LOW BYTES FOR WORD MOVE
5408 00002A5E F6C280               <1>     test    dl, 80h                 ; AGAIN, IS THIS XOR FUNCTION
5409 00002A61 7407                 <1>     jz      short _S10              ; NO, JUST STORE THE VALUES
5410 00002A63 66338700200000       <1>     xor     ax, [edi+2000h]                 ; FUNCTION WITH FIRST HALF LOW
5411                                <1> _S10:
5412 00002A6A 66898700200000       <1>     mov     [edi+2000h], ax                 ; STORE SECOND PORTION HIGH
5413 00002A71 6683C750             <1>     add     di, 80                  ; POINT TO NEXT LOCATION
5414 00002A75 FECE                 <1>     dec     dh
5415 00002A77 75C4                 <1>     jnz     short S8                ; KEEP GOING
5416 00002A79 5E                   <1>     pop     esi                     ; RECOVER CODE POINTER
5417 00002A7A 5F                   <1>     pop     edi                     ; RECOVER REGEN POINTER
5418 00002A7B 47                   <1>     inc     edi                     ; POINT TO NEXT CHAR POSITION
5419 00002A7C 47                   <1>     inc     edi
5420 00002A7D E2BA                 <1>     loop    S7                      ; MORE TO WRITE
5421 00002A7F C3                   <1>     retn
5422                                <1>
5423                                <1> ; 03/08/2022 - TRDOS 386 Kernel v2.0.5
5424                                <1> ; 04/07/2016
5425                                <1> ; 01/07/2016
5426                                <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
5427                                <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
5428                                <1> ;----------------------------------------
5429                                <1> ; GRAPHICS READ
5430                                <1> ;----------------------------------------
5431                                <1> GRAPHICS_READ:
5432 00002A80 E89E000000           <1>     call    S26                     ; CONVERTED TO OFFSET IN REGEN
5433 00002A85 89C6                 <1>     mov     esi, eax                ; SAVE IN SI
5434 00002A87 81C600800B00         <1>     add     esi, 0B8000h            ; 01/07/2016
5435 00002A8D 83EC08               <1>     sub     esp, 8                  ; ALLOCATE SPACE FOR THE READ CODE POINT
5436 00002A90 89E5                 <1>     mov     ebp, esp                ; POINTER TO SAVE AREA
5437                                <1>
5438                                <1> ;-----      DETERMINE GRAPHICS MODES
5439 00002A92 B604                 <1>     mov     dh, 4                   ; number of passes ; 01/07/2016
5440 00002A94 803D[1E680000]06     <1>     cmp     byte [CRT_MODE], 6
5441 00002A9B 7219                 <1>     jc      short S12               ; MEDIUM RESOLUTION
5442                                <1>
5443                                <1> ;-----      HIGH RESOLUTION READ
5444                                <1> ;-----      GET VALUES FROM REGEN BUFFER AND CONVERT TO CODE POINT
5445                                <1>     ;mov    dh,4                    ; NUMBER OF PASSES
5446                                <1> S11:
5447 00002A9D 8A06                 <1>     mov     al, [esi]               ; GET FIRST BYTE
5448 00002A9F 884500               <1>     mov     [ebp], al               ; SAVE IN STORAGE AREA
5449 00002AA2 45                   <1>     inc     ebp                     ; NEXT LOCATION
5450 00002AA3 8A8600200000         <1>     mov     al, [esi+2000h]                 ; GET LOWER REGION BYTE
5451 00002AA9 884500               <1>     mov     [ebp], al               ; ADJUST AND STORE
5452 00002AAC 45                   <1>     inc     ebp
5453 00002AAD 83C650               <1>     add     esi, 80                 ; POINTER INTO REGEN
5454 00002AB0 FECE                 <1>     dec     dh                      ; LOOP CONTROL
5455 00002AB2 75E9                 <1>     jnz     short S11               ; DO IT SOME MORE
5456 00002AB4 EB1C                 <1>     jmp     short S14               ; GO MATCH THE SAVED CODE POINTS
5457                                <1>
5458                                <1> ;-----      MEDIUM RESOLUTION READ
5459                                <1> S12:
5460                                <1>     ;sal    si, 1                   ; OFFSET*2 SINCE 2 BYTES/CHAR
5461                                <1>     ; 03/08/2022
5462 00002AB6 D1E6                 <1>     sal     esi, 1
5463                                <1>     ;mov    dh, 4                   ; NUMBER OF PASSES
5464                                <1> S13:
5465 00002AB8 E84B000000           <1>     call    S23                     ; GET BYTES FROM REGEN INTO SINGLE SAVE
5466 00002ABD 81C6FE1F0000         <1>     add     esi, 2000h-2            ; GO TO LOWER REGION
5467 00002AC3 E840000000           <1>     call    S23                     ; GET THIS PAIR INTO SAVE
5468 00002AC8 81EEB21F0000         <1>     sub     esi, 2000h-80+2                 ; ADJUST POINTER BACK INTO UPPER
5469 00002ACE FECE                 <1>     dec     dh
5470 00002AD0 75E6                 <1>     jnz     short S13               ; KEEP GOING UNTIL ALL 8 DONE
5471                                <1>
5472                                <1> ;-----      SAVE AREA HAS CHARACTER IN IT, MATCH IT
5473                                <1> S14:                                        ; FIND_CHAR
5474 00002AD2 BF[104D0100]         <1>     mov     edi, CRT_CHAR_GEN       ; ESTABLISH ADDRESSING
5475 00002AD7 83ED08               <1>     sub     ebp, 8                  ; ADJUST POINTER TO START OF SAVE AREA
5476 00002ADA 89EE                 <1>     mov     esi, ebp
5477                                <1> S15:
5478                                <1>     ;mov    ax, 256                 ; NUMBER TO TEST AGAINST
5479                                <1>     ; 03/08/2022
5480 00002ADC 29C0                 <1>     sub     eax, eax
5481 00002ADE FEC4                 <1>     inc     ah
5482                                <1>     ; eax = 256
5483                                <1> S16:
5484 00002AE0 56                   <1>     push    esi                     ; SAVE SAVE AREA POINTER
5485 00002AE1 57                   <1>     push    edi                     ; SAVE CODE POINTER
5486                                <1>     ;mov    ecx, 4                  ; NUMBER OF WORDS TO MATCH
5487                                <1>     ;repe   cmpsw                   ; COMPARE THE 8 BYTES AS WORDS
5488 00002AE2 A7                   <1>     cmpsd                           ; compare first 4 bytes
5489 00002AE3 7501                 <1>     jne     short S17               ;
5490 00002AE5 A7                   <1>     cmpsd                           ; compare last 4 bytes
5491                                <1> S17:
5492 00002AE6 5F                   <1>     pop     edi                     ; RECOVER THE POINTERS
5493 00002AE7 5E                   <1>     pop     esi
5494                                <1>     ;jz     short S18               ; IF ZERO FLAG SET, THEN MATCH OCCURRED
5495 00002AE8 7406                 <1>     je      short S18
5496                                <1>     ;                               ; NO MATCH, MOVE ON TO NEXT
5497 00002AEA 83C708               <1>     add     edi, 8                  ; NEXT CODE POINT
5498                                <1>     ;dec    ax                      ; LOOP CONTROL
5499                                <1>     ; 03/08/2022
5500 00002AED 48                   <1>     dec     eax
5501 00002AEE 75F0                 <1>     jnz     short S16               ; DO ALL OF THEM
5502                                <1>
5503                                <1> ;-----      CHARACTER IS FOUND ( AL=0 IF NOT FOUND )
5504                                <1> S18:
5505 00002AF0 83C408               <1>     add     esp, 8                  ; READJUST THE STACK, THROW AWAY SAVE
```

```
5506 00002AF3 C3                <1>     retn                            ; ALL DONE
5507                            <1>
5508                            <1> ; 12/04/2021
5509                            <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
5510                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
5511                            <1> ;--------------------------------------------
5512                            <1> ; EXPAND BYTE
5513                            <1> ;  THIS ROUTINE TAKES THE BYTE IN AL AND DOUBLES ALL
5514                            <1> ;  OF THE BITS, TURNING THE 8 BITS INTO 16 BITS.
5515                            <1> ;  THE RESULT IS LEFT IN AX
5516                            <1> ;--------------------------------------------
5517                            <1> S21:
5518                            <1>     ; 03/08/2022
5519                            <1>     ;push   cx                      ; SAVE REGISTER
5520                            <1>     ; 12/04/2021
5521 00002AF4 51                <1>     push    ecx
5522                            <1>     ;;mov   cx, 8                   ; SHIFT COUNT REGISTER FOR ONE BYTE
5523                            <1>     ;mov    cl, 8
5524 00002AF5 B408              <1>     mov     ah, 8
5525                            <1> S22:
5526 00002AF7 D0C8              <1>     ror     al, 1                   ; SHIFT BITS, LOW BIT INTO CARRY FLAG
5527                            <1>     ;rcr    bp, 1                   ; MOVE CARRY FLAG (LOW BIT INTO RESULTS
5528                            <1>     ;sar    bp, 1                   ; SIGN EXTEND HIGH BIT (DOUBLE IT)
5529                            <1>     ; 03/08/2022
5530 00002AF9 66D1D9            <1>     rcr     cx, 1
5531 00002AFC 66D1F9            <1>     sar     cx, 1
5532                            <1>
5533                            <1>     ;;loop  S22                     ; REPEAT FOR ALL 8 BITS
5534                            <1>     ;dec    cl
5535                            <1>     ;jnz    short S22
5536                            <1>     ;xchg   ax, bp                  ; MOVE RESULTS TO PARAMETER REGISTER
5537                            <1>     ; 03/08/5022
5538 00002AFF FECC              <1>     dec     ah
5539 00002B01 75F4              <1>     jnz     short S22
5540 00002B03 6689C8            <1>     mov     ax, cx
5541                            <1>     ;pop    cx                      ; RECOVER REGISTER
5542                            <1>     ; 12/04/2021
5543 00002B06 59                <1>     pop     ecx
5544 00002B07 C3                <1>     retn                            ; ALL DONE
5545                            <1>
5546                            <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
5547                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
5548                            <1> ;--------------------------------------------------
5549                            <1> ; MED_READ_BYTE
5550                            <1> ; THIS ROUTINE WILL TAKE 2 BYTES FROM THE REGEN BUFFER,
5551                            <1> ;  COMPARE AGAINST THE CURRENT FOREGROUND COLOR, AND PLACE
5552                            <1> ;  THE CORRESPONDING ON/OFF BIT PATTERN INTO THE CURRENT
5553                            <1> ;  POSITION IN THE SAVE AREA
5554                            <1> ; ENTRY --
5555                            <1> ;  SI,DS = POINTER TO REGEN AREA OF INTEREST
5556                            <1> ;  BX = EXPANDED FOREGROUND COLOR
5557                            <1> ;  BP = POINTER TO SAVE AREA
5558                            <1> ; EXIT --
5559                            <1> ;  SI AND BP ARE INCREMENTED
5560                            <1> ;--------------------------------------------------
5561                            <1> S23:
5562 00002B08 66AD              <1>     lodsw                           ; GET FIRST BYTE AND SECOND BYTES
5563 00002B0A 86E0              <1>     xchg    al, ah                  ; SWAP FOR COMPARE
5564                            <1>     ;mov    cx, 0C000h              ; 2 BIT MASK TO TEST THE ENTRIES
5565                            <1>     ; 02/08/2022
5566 00002B0C 29C9              <1>     sub     ecx, ecx
5567 00002B0E B5C0              <1>     mov     ch, 0C0h
5568                            <1>     ; ecx = 0C000h
5569                            <1>     ;mov    dl, 0                   ; RESULT REGISTER
5570                            <1>     ; 03/08/2022
5571 00002B10 28D2              <1>     sub     dl, dl
5572                            <1> S24:
5573                            <1>     ;test   ax, cx                  ; IS THIS SECTION BACKCROUND?
5574                            <1>         ; 03/08/2022
5575 00002B12 85C8              <1>     test    eax, ecx
5576 00002B14 7401              <1>     jz      short S25               ; IF ZERO, IT IS BACKGROUND (CARRY=0)
5577 00002B16 F9                <1>     stc                             ; WASN'T, SO SET CARRY
5578                            <1> S25:
5579 00002B17 D0D2              <1>     rcl     dl, 1                   ; MOVE THAT BIT INTO THE RESULT
5580                            <1>     ;shr    cx, 2                   ; MOVE THE MASK TO THE RIGHT BY 2 BITS
5581                            <1>     ; 02/08/2022
5582 00002B19 C1E902            <1>     shr     ecx, 2
5583 00002B1C 73F4              <1>     jnc     short S24               ; DO IT AGAIN IF MASK DIDN'T FALL OUT
5584 00002B1E 885500            <1>     mov     [ebp], dl               ; STORE RESULT IN SAVE AREA
5585 00002B21 45                <1>     inc     ebp                     ; ADJUST POINTER
5586 00002B22 C3                <1>     retn                            ; ALL DONE
5587                            <1>
5588                            <1> ; 02/08/2022 - TRDOS 386 Kernel v2.0.5
5589                            <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
5590                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
5591                            <1> ;--------------------------------------------
5592                            <1> ; V4_POSITION
5593                            <1> ;  THIS ROUTINE TAKES THE CURSOR POSITION CONTAINED IN
5594                            <1> ;  THE MEMORY LOCATION, AND CONVERTS IT INTO AN OFFSET
5595                            <1> ;  INTO THE REGEN BUFFER, ASSUMING ONE BYTE/CHAR.
5596                            <1> ;  FOR MEDIUM RESOLUTION GRAPHICS, THE NUMBER MUST
5597                            <1> ;  BE DOUBLED.
5598                            <1> ; ENTRY -- NO REGISTERS,MEMORY LOCATION @CURSOR_POSN IS USED
5599                            <1> ; EXIT--
5600                            <1> ;  AX CONTAINS OFFSET INTO REGEN BUFFER
5601                            <1> ;--------------------------------------------
5602                            <1> S26:
5603 00002B23 0FB705[9E760100]  <1>     movzx   eax, word [CURSOR_POSN]       ; GET CURRENT CURSOR
5604                            <1> GRAPH_POSN:
5605 00002B2A 53                <1>     push    ebx                     ; SAVE REGISTER
5606 00002B2B 0FB6D8            <1>     movzx   ebx, al                 ; SAVE A COPY OF CURRENT CURSOR
5607 00002B2E A0[20680000]      <1>     mov     al, [CRT_COLS]          ; GET BYTES PER COLUMN
5608 00002B33 F6E4              <1>     mul     ah                      ; MULTIPLY BY ROWS
5609                            <1>     ;shl    ax, 2                   ; MULTIPLY * 4 SINCE 4 ROWS/BYTE
5610                            <1>     ; 02/08/2022
5611 00002B35 C1E002            <1>     shl     eax, 2
5612 00002B38 01D8              <1>     add     eax, ebx                ; DETERMINE OFFSET
5613 00002B3A 5B                <1>     pop     ebx                     ; RECOVER POINTER
5614 00002B3B C3                <1>     retn                            ; ALL DONE
5615                            <1>
5616                            <1> ; 03/08/2022 - TRDOS 386 Kernel v2.0.5
5617                            <1> ; 09/07/2016
5618                            <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
5619                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
5620                            <1> ;--------------------------------------------
5621                            <1> ; SET_COLOR
5622                            <1> ;   THIS ROUTINE WILL ESTABLISH THE BACKGROUND COLOR, THE OVERSCAN COLOR,
5623                            <1> ;   AND THE FOREGROUND COLOR SET FOR MEDIUM RESOLUTION GRAPHICS
5624                            <1> ; INPUT
5625                            <1> ;   (BH) HAS COLOR ID
5626                            <1> ;           IF BH=0, THE BACKGROUND COLOR VALUE IS SET
5627                            <1> ;                   FROM THE LOW BITS OF BL (0-31)
5628                            <1> ;           IF BH=1, THE PALETTE SELECTION IS MADE
5629                            <1> ;                   BASED ON THE LOW BIT OF BL:
```

```
                                <1> ;                       0 = GREEN, RED, YELLOW FOR COLORS 1,2,3
                                <1> ;                       1 = BLUE, CYAN, MAGENTA FOR COLORS 1,2,3
                                <1> ;   (BL) HAS THE COLOR VALUE TO BE USED
                                <1> ; OUTPUT
                                <1> ;   THE COLOR SELECTION IS UPDATED
                                <1> ;-----------------------------------------------
                                <1> SET_COLOR:
00002B3C 803D[1E680000]07       <1>         cmp     byte [CRT_MODE], 7      ; 09/07/2016
                                <1>     ;ja     VIDEO_RETURN            ; nothing to do for VGA modes
                                <1>     ; 03/08/2022
00002B43 7605                   <1>     jna     short M21
00002B45 E92EF0FFFF             <1>     jmp     VIDEO_RETURN
                                <1> M21:
                                <1>     ;mov    dx, [ADDR_6845]             ; I/O PORT FOR PALETTE
                                <1>     ;mov    dx, 3D4h
                                <1>     ;add    dx, 5                   ; OVERSCAN PORT
00002B4A 66BAD903               <1>     mov     dx, 3D9h
00002B4E A0[21680000]           <1>     mov     al, [CRT_PALETTE]       ; GET THE CURRENT PALETTE VALUE
00002B53 08FF                   <1>     or      bh, bh                  ; IS THIS COLOR 0?
00002B55 7512                   <1>     jnz     short M20               ; OUTPUT COLOR 1
                                <1>
                                <1> ;-----      HANDLE COLOR 0 BY SETTING THE BACKGROUND COLOR
                                <1>
00002B57 24E0                   <1>     and     al, 0E0h                ; TURN OFF LOW 5 BITS OF CURRENT
00002B59 80E31F                 <1>     and     bl, 01Fh                ; TURN OFF HIGH 3 BITS OF INPUT VALUE
00002B5C 08D8                   <1>     or      al, bl                  ; PUT VALUE INTO REGISTER
                                <1> M19:                                ; OUTPUT THE PALETTE
00002B5E EE                     <1>     out     dx, al                  ; OUTPUT COLOR SELECTION TO 3D9 PORT
00002B5F A2[21680000]           <1>     mov     [CRT_PALETTE], al       ; SAVE THE COLOR VALUE
00002B64 E90FF0FFFF             <1>     jmp     VIDEO_RETURN
                                <1>
                                <1> ;-----      HANDLE COLOR 1 BY SELECTING THE PALETTE TO BE USED
                                <1>
                                <1> M20:
00002B69 24DF                   <1>     and     al, 0DFH                ; TURN OFF PALETTE SELECT BIT
00002B6B D0EB                   <1>     shr     bl, 1                   ; TEST THE LOW ORDER BIT OF BL
00002B6D 73EF                   <1>     jnc     short M19               ; ALREADY DONE
00002B6F 0C20                   <1>     or      al, 20H                 ; TURN ON PALETTE SELECT BIT
00002B71 EBEB                   <1>     jmp     short M19               ; GO DO IT
                                <1>
                                <1> ; 03/08/2022 - TRDOS 386 Kernel v2.0.5
                                <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
00002B73 8A25[1E680000]         <1>     mov     ah,  [CRT_MODE]
00002B79 80FC07                 <1>     cmp     ah, 7 ; 6!?
00002B7C 760A                   <1>     jna     short read_dot_cga
                                <1>
00002B7E E8B3030000             <1>     call    vga_read_pixel
                                <1>     ; al = pixel value
                                <1> read_dot_retn:      ; 03/08/2022
00002B83 E9F5EFFFFF             <1>     jmp     _video_return
                                <1>
                                <1> read_dot_cga:
                                <1>     ;je     VIDEO_RETURN ; 7
00002B88 80FC04                 <1>     cmp     ah, 4 ; graphics ?
                                <1>     ;jb     VIDEO_RETURN ; no, text mode, nothing to do
                                <1>     ; 03/08/2022
00002B8B 72F6                   <1>     jb      short read_dot_retn
                                <1>
00002B8D E84F000000             <1>     call    R3                      ; DETERMINE BYTE POSITION OF DOT
00002B92 8A06                   <1>     mov     al, [esi]               ; GET THE BYTE
00002B94 20E0                   <1>     and     al, ah                  ; MASK OFF THE OTHER BITS IN THE BYTE
00002B96 D2E0                   <1>     shl     al, cl                  ; LEFT JUSTIFY THE VALUE
00002B98 88F1                   <1>     mov     cl, dh                  ; GET NUMBER OF BITS IN RESULT
00002B9A D2C0                   <1>     rol     al, cl                  ; RIGHT JUSTIFY THE RESULT
                                <1>     ;jmp    VIDEO_RETURN            ; RETURN FROM VIDEO I/O
00002B9C 0FB6C0                 <1>     movzx   eax, al
00002B9F E9D9EFFFFF             <1>     jmp     _video_return
                                <1>
                                <1> ; 03/08/2022
                                <1> ; 02/08/2022 - TRDOS 386 Kernel v2.0.5
                                <1> ; 12/04/2021
                                <1> ; 09/07/2016
                                <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
                                <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
                                <1>
                                <1> WRITE_DOT:
                                <1>     ; 09/07/2016
00002BA4 8A25[1E680000]         <1>     mov     ah, [CRT_MODE]
00002BAA 80FC07                 <1>     cmp     ah, 7 ; 6!?
00002BAD 760A                   <1>     jna     short write_dot_cga
                                <1>
00002BAF E8F2020000             <1>     call    vga_write_pixel
                                <1> write_dot_retn:     ; 03/08/2022
00002BB4 E9BFEFFFFF             <1>     jmp     VIDEO_RETURN
                                <1>
                                <1> write_dot_cga:
                                <1>     ;je     VIDEO_RETURN ; 7
00002BB9 80FC04                 <1>     cmp     ah, 4 ; graphics ?
                                <1>     ;jb     VIDEO_RETURN ; no, text mode, nothing to do
                                <1>     ; 03/08/2022
00002BBC 72F6                   <1>     jb      short write_dot_retn
                                <1>
                                <1>     ;;push  ax                      ; SAVE DOT VALUE
                                <1>     ;push   ax                      ; TWICE
                                <1>     ; 12/04/2021
00002BBE 50                     <1>     push    eax
00002BBF E81D000000             <1>     call    R3                      ; DETERMINE BYTE POSITION OF THE DOT
00002BC4 D2E8                   <1>     shr     al, cl                  ; SHIFT TO SET UP THE BITS FOR OUTPUT
00002BC6 20E0                   <1>     and     al, ah                  ; STRIP OFF THE OTHER BITS
00002BC8 8A0E                   <1>     mov     cl, [esi]               ; GET THE CURRENT BYTE
                                <1>     ;pop    bx                      ; RECOVER XOR FLAG
                                <1>     ; 12/04/2021
00002BCA 5B                     <1>     pop     ebx
```

```
5754 00002BCB F6C380            <1>     test    bl, 80h                 ; IS IT ON
5755 00002BCE 750D              <1>     jnz     short R2                ; YES, XOR THE DOT
5756 00002BD0 F6D4              <1>     not     ah                      ; SET MASK TO REMOVE THE INDICATED BITS
5757 00002BD2 20E1              <1>     and     cl, ah
5758 00002BD4 08C8              <1>     or      al, cl                  ; OR IN THE NEW VALUE OF THOSE BITS
5759                            <1> R1:                                 ; FINISH_DOT
5760 00002BD6 8806              <1>     mov     [esi], al               ; RESTORE THE BYTE IN MEMORY
5761                            <1>     ;;pop   AX
5762 00002BD8 E99BEFFFFF        <1>     jmp     VIDEO_RETURN            ; RETURN FROM VIDEO I/O
5763                            <1> R2:                                 ; XOR_DOT
5764 00002BDD 30C8              <1>     xor     al, cl                  ; EXCLUSIVE OR THE DOTS
5765 00002BDF EBF5              <1>     jmp     short R1                ; FINISH UP THE WRITING
5766                            <1>
5767                            <1> ; 02/08/2022 - TRDOS 386 Kernel v2.0.5
5768                            <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
5769                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
5770                            <1>
5771                            <1> ;-----------------------------------------------
5772                            <1> ; THIS SUBROUTINE DETERMINES THE REGEN BYTE LOCATION OF THE
5773                            <1> ; INDICATED ROW COLUMN VALUE IN GRAPHICS MODE.
5774                            <1> ; ENTRY --
5775                            <1> ;  DX = ROW VALUE (0-199)
5776                            <1> ;  CX = COLUMN VALUE (0-639)
5777                            <1> ; EXIT --
5778                            <1> ;  SI = OFFSET INTO REGEN BUFFER FOR BYTE OF INTEREST
5779                            <1> ;  AH = MASK TO STRIP OFF THE BITS OF INTEREST
5780                            <1> ;  CL = BITS TO SHIFT TO RIGHT JUSTIFY THE MASK IN AH
5781                            <1> ;  DH = # BITS IN RESULT
5782                            <1> ;  BX = MODIFIED
5783                            <1> ;-----------------------------------------------
5784                            <1> R3:
5785                            <1>
5786                            <1> ;-----      DETERMINE 1ST BYTE IN INDICATED ROW BY MULTIPLYING ROW VALUE BY 40
5787                            <1> ;-----       ( LOW BIT OF ROW DETERMINES EVEN/ODD, 80 BYTES/ROW )
5788                            <1>
5789 00002BE1 0FB7F0            <1>     movzx   esi, ax                 ; WILL SAVE AL AND AH DURING OPERATION
5790 00002BE4 B028              <1>     mov     al, 40
5791 00002BE6 F6E2              <1>     mul     dl                      ; AX= ADDRESS OF START OF INDICATED ROW
5792 00002BE8 A808              <1>     test    al, 08H                 ; TEST FOR EVEN/ODD ROW CALCULATED
5793 00002BEA 7404              <1>     JZ      short R4                ; JUMP IF EVEN ROW
5794 00002BEC 6605D81F          <1>     add     ax, 2000h-40            ; OFFSET TO LOCATION OF ODD ROWS ADJUST
5795                            <1> R4:                                 ; EVEN_ROW
5796                            <1>     ;xchg   si, ax                  ; MOVE POINTER TO (SI) AND RECOVER (AX)
5797                            <1>     ; 02/08/2022
5798 00002BF0 96                <1>     xchg    esi, eax
5799 00002BF1 81C600800B00      <1>     add     esi, 0B8000h
5800                            <1>     ;mov    dx, cx                  ; COLUMN VALUE TO DX
5801 00002BF7 0FB7D1            <1>     movzx   edx, cx
5802                            <1>
5803                            <1> ;-----      DETERMINE GRAPHICS MODE CURRENTLY IN EFFECT
5804                            <1>
5805                            <1> ; SET UP THE REGISTERS ACCORDING TO THE MODE
5806                            <1> ; CH = MASK FOR LOW OF COLUMN ADDRESS ( 7/3 FOR HIGH/MED RES )
5807                            <1> ; CL = # OF ADDRESS BITS IN COLUMN VALUE ( 3/2 FOR H/M )
5808                            <1> ; BL = MASK TO SELECT BITS FROM POINTED BYTE ( 80H/C0H FOR H/M )
5809                            <1> ; BH = NUMBER OF VALID BITS IN POINTED BYTE ( 1/2 FOR H/M )
5810                            <1>
5811 00002BFA 66BBC002          <1>     mov     bx, 2C0h
5812 00002BFE 66B90203          <1>     mov     cx, 302h                ; SET PARMS FOR MED RES
5813 00002C02 803D[1E680000]06  <1>     cmp     byte [CRT_MODE], 6
5814 00002C09 7208              <1>     jc      short R5                ; HANDLE IF MED RES
5815 00002C0B 66BB8001          <1>     mov     bx, 180h
5816 00002C0F 66B90307          <1>     mov     cx, 703h                ; SET PARMS FOR HIGH RES
5817                            <1>
5818                            <1> ;-----      DETERMINE BIT OFFSET IN BYTE FROM COLUMN MASK
5819                            <1> R5:
5820 00002C13 20D5              <1>     and     ch, dl                  ; ADDRESS OF PEL WITHIN BYTE TO CH
5821                            <1>
5822                            <1> ;-----      DETERMINE BYTE OFFSET FOR THIS LOCATION IN COLUMN
5823                            <1>
5824                            <1>     ;shr    dx, cl                  ; SHIFT BY CORRECT AMOUNT
5825                            <1>     ;add    si, dx                  ; INCREMENT THE POINTER
5826                            <1>     ; 02/08/2022
5827 00002C15 D3EA              <1>     shr     edx, cl
5828 00002C17 01D6              <1>     add     esi, edx
5829 00002C19 88FE              <1>     mov     dh, bh                  ; GET THE # OF BITS IN RESULT TO DH
5830                            <1>
5831                            <1> ;-----      MULTIPLY BH (VALID BITS IN BYTE) BY CH (BIT OFFSET)
5832                            <1>
5833 00002C1B 28C9              <1>     sub     cl, cl                  ; ZERO INTO STORAGE LOCATION
5834                            <1> R6:
5835 00002C1D D0C8              <1>     ror     al, 1                   ; LEFT JUSTIFY VALUE IN AL (FOR WRITE)
5836 00002C1F 00E9              <1>     add     cl, ch                  ; ADD IN THE BIT OFFSET VALUE
5837 00002C21 FECF              <1>     dec     bh                      ; LOOP CONTROL
5838 00002C23 75F8              <1>     jnz     short R6                ; ON EXIT, CL HAS COUNT TO RESTORE BITS
5839 00002C25 88DC              <1>     mov     ah, bl                  ;  GET MASK TO AH
5840 00002C27 D2EC              <1>     shr     ah, cl                  ;  MOVE THE MASK TO CORRECT LOCATION
5841 00002C29 C3                <1>     retn                            ;  RETURN WITH EVERYTHING SET UP
5842                            <1>
5843                            <1> load_dac_palette:
5844                            <1>     ; 02/08/2022 (TRDOS 386 Kernel v2.0.5)
5845                            <1>     ; 29/07/2016
5846                            <1>     ; 23/07/2016
5847                            <1>     ; 03/07/2016 (TRDOS 386 = TRDOS v2.0)
5848                            <1>     ; (set_mode_vga)
5849                            <1>     ; derived from 'Plex86/Bochs VGABios' source code
5850                            <1>     ; vgabios-0.7a (2011)
5851                            <1>     ; by the LGPL VGABios developers Team (2001-2008)
5852                            <1>     ; 'vgabios.c', 'load_dac_palette'
5853                            <1>     ;
5854                            <1>     ; Oracle VirtualBox 5.0.24 VGABios Source Code
5855                            <1>     ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
5856                            <1>     ;
5857                            <1>     ; INPUT -> AH = DAC selection number (3, 2 or 1)
5858                            <1>     ; OUTPUT -> ECX = 0, AX = 0
5859                            <1>     ; (Modifed registers: EAX, ECX, EDX, ESI)
5860                            <1>     ;
5861 00002C2A 66BAC803          <1>     mov     dx, 3C8h  ; VGAREG_DAC_WRITE_ADDRESS
5862 00002C2E 28C0              <1>     sub     al, al ; 0
5863 00002C30 EE                <1>     out     dx, al ; 0 ; color index, always 0 at the beginning
5864                            <1>     ;inc    dx   ; 3C9h ; VGAREG_DAC_DATA
5865                            <1>     ; 02/08/2022
5866 00002C31 FEC2              <1>     inc     dl ; dx = 3C9h
5867                            <1>     ;mov    ecx, 256   ; always 256*3 values
5868                            <1>     ; 02/08/2022
5869 00002C33 31C9              <1>     xor     ecx, ecx
5870 00002C35 FEC5              <1>     inc     ch
5871                            <1>     ; ecx = 256
5872                            <1>
5873                            <1>     ;push   esi
5874 00002C37 88E0              <1>     mov     al, ah
5875 00002C39 B43F              <1>     mov     ah, 3Fh ; 3Fh except DAC selection number 3
5876 00002C3B 3C02              <1>     cmp     al, 2
5877 00002C3D 7414              <1>     je      short l_dac_p_2
```

```
5878 00002C3F 7719              <1>     ja      short l_dac_p_3
5879 00002C41 20C0              <1>     and     al, al
5880 00002C43 7507              <1>     jnz     short l_dac_p_1
5881                            <1> l_dac_p_0:
5882 00002C45 BE[D0470100]      <1>     mov     esi, palette0
5883 00002C4A EB15              <1>     jmp     short l_dac_p_4
5884                            <1> l_dac_p_1:
5885 00002C4C BE[90480100]      <1>     mov     esi, palette1
5886 00002C51 EB0E              <1>     jmp     short l_dac_p_4
5887                            <1> l_dac_p_2:
5888 00002C53 BE[50490100]      <1>     mov     esi, palette2
5889 00002C58 EB07              <1>     jmp     short l_dac_p_4
5890                            <1> l_dac_p_3:
5891 00002C5A B4FF              <1>     mov     ah, 0FFh ; dac registers
5892 00002C5C BE[104A0100]      <1>     mov     esi, palette3
5893                            <1> l_dac_p_4:
5894 00002C61 AC                <1>     lodsb
5895 00002C62 EE                <1>     out     dx, al  ; Red
5896 00002C63 AC                <1>     lodsb
5897 00002C64 EE                <1>     out     dx, al ; Green
5898 00002C65 AC                <1>     lodsb
5899 00002C66 EE                <1>     out     dx, al ; Blue
5900 00002C67 20E4              <1>     and     ah, ah
5901 00002C69 7405              <1>     jz      short l_dac_p_5
5902 00002C6B FECC              <1>     dec     ah
5903 00002C6D E2F2              <1>     loop    l_dac_p_4
5904                            <1>     ;pop    esi
5905 00002C6F C3                <1>     retn
5906                            <1> l_dac_p_5:
5907                            <1>     ; 29/07/2016
5908 00002C70 FEC9              <1>     dec     cl
5909 00002C72 7407              <1>     jz      short l_dac_p_7
5910                            <1>     ;
5911 00002C74 28C0              <1>     sub     al, al ; 0
5912                            <1> l_dac_p_6:
5913 00002C76 EE                <1>     out     dx, al ; outb(VGAREG_DAC_DATA,0);
5914 00002C77 EE                <1>     out     dx, al
5915 00002C78 EE                <1>     out     dx, al
5916 00002C79 E2FB              <1>     loop    l_dac_p_6
5917                            <1> l_dac_p_7:
5918                            <1>     ;pop    esi
5919 00002C7B C3                <1>     retn
5920                            <1>
5921                            <1> gray_scale_summing:
5922                            <1>     ; 03/08/2022 (TRDOS 386 v2.0.5)
5923                            <1>     ; 12/04/2021
5924                            <1>     ; 03/07/2016 (TRDOS 386 = TRDOS v2.0)
5925                            <1>     ; (set_mode_vga)
5926                            <1>     ; derived from 'Plex86/Bochs VGABios' source code
5927                            <1>     ; vgabios-0.7a (2011)
5928                            <1>     ; by the LGPL VGABios developers Team (2001-2008)
5929                            <1>     ; 'vgabios.c', 'biosfn_perform_gray_scale_summing'
5930                            <1>     ;
5931                            <1>     ; Oracle VirtualBox 5.0.24 VGABios Source Code
5932                            <1>     ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
5933                            <1>     ;
5934                            <1>
5935                            <1>     ; INPUT -> EBX = Start address (color index <= 255)
5936                            <1>     ;         ECX = Count (<= 256)
5937                            <1>     ; OUTPUT -> (E)CX = 0
5938                            <1>     ; (Modifed registers: EAX, ECX, EDX, EBX)
5939                            <1>
5940 00002C7C 66BADA03          <1>     mov     dx, 3DAh ; VGAREG_ACTL_RESET
5941 00002C80 EC                <1>     in      al, dx
5942 00002C81 30C0              <1>     xor     al, al ; 0
5943                            <1>     ;mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
5944                            <1>     ; 03/08/2022
5945 00002C83 B2C0              <1>     mov     dl, 0C0h
5946 00002C85 EE                <1>     out     dx, al  ; clear bit 5
5947                            <1>                     ; (while loading palette registers)
5948                            <1>     ; set read address and switch to read mode
5949                            <1> g_s_s_1:
5950                            <1>     ;mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
5951                            <1>     ; 03/08/2022
5952 00002C86 B2C7              <1>     mov     dl, 0C7h
5953 00002C88 88D8              <1>     mov     al, bl
5954 00002C8A EE                <1>     out     dx, al
5955                            <1>     ; get 6-bit wide RGB data values
5956                            <1>     ; intensity = (0.3*Red)+(0.59*Green)+(0.11*Blue)
5957                            <1>     ; i = ( ( 77*r + 151*g + 28*b ) + 0x80 ) >> 8;
5958                            <1>     ;mov    dx, 3C9h ; VGAREG_DAC_DATA
5959                            <1>     ; 03/08/2022
5960 00002C8B B2C9              <1>     mov     dl, 0C9h
5961 00002C8D EC                <1>     in      al, dx ; red
5962 00002C8E B44D              <1>     mov     ah, 77 ; 0.3* Red
5963 00002C90 F6E4              <1>     mul     ah
5964                            <1>     ;push   ax
5965                            <1>     ; 12/04/2021
5966 00002C92 50                <1>     push    eax
5967 00002C93 EC                <1>     in      al, dx ; green
5968 00002C94 B497              <1>     mov     ah, 151  ; 0.59 * Green
5969 00002C96 F6E4              <1>     mul     ah
5970                            <1>     ;push   ax
5971                            <1>     ; 12/04/2021
5972 00002C98 50                <1>     push    eax
5973 00002C99 EC                <1>     in      al, dx ; blue
5974 00002C9A B41C              <1>     mov     ah, 28 ; 0.11 * Blue
5975 00002C9C F6E4              <1>     mul     ah
5976                            <1>     ;pop    dx
5977                            <1>     ; 12/04/2021
5978 00002C9E 5A                <1>     pop     edx
5979 00002C9F 6601D0            <1>     add     ax, dx
5980                            <1>     ;pop    dx
5981                            <1>     ; 12/04/2021
5982 00002CA2 5A                <1>     pop     edx
5983 00002CA3 6601D0            <1>     add     ax, dx
5984 00002CA6 66058000          <1>     add     ax, 80h
5985 00002CAA B03F              <1>     mov     al, 3Fh
5986 00002CAC 38C4              <1>     cmp     ah, al  ; if(i>0x3f)i=0x3f
5987 00002CAE 7602              <1>     jna     short g_s_s_2
5988 00002CB0 88C4              <1>     mov     ah, al
5989                            <1> g_s_s_2:
5990                            <1>     ;mov    dx, 3C8h  ; VGAREG_DAC_WRITE_ADDRESS
5991                            <1>     ; 03/08/2022
5992 00002CB2 B2C8              <1>     mov     dl, 0C8h
5993 00002CB4 88D8              <1>     mov     al, bl ; color index
5994 00002CB6 EE                <1>     out     dx, al
5995 00002CB7 88E0              <1>     mov     al, ah ; intensity
5996                            <1>     ;inc    dx ; 3C9h ; VGAREG_DAC_DATA
5997                            <1>     ; 03/08/2022
5998 00002CB9 FEC2              <1>     inc     dl
5999 00002CBB EE                <1>     out     dx, al ; R (R=G=B)
6000 00002CBC 88E0              <1>     mov     al, ah ; intensity
6001 00002CBE EE                <1>     out     dx, al ; G (R=G=B)
```

```
6002 00002CBF 88E0              <1>     mov     al, ah ; intensity
6003 00002CC1 EE                <1>     out     dx, al ; B (R=G=B)
6004                            <1>     ;dec    cx
6005                            <1>     ; 03/08/2022
6006 00002CC2 49                <1>     dec     ecx
6007 00002CC3 7404              <1>     jz      short g_s_s_3
6008 00002CC5 FEC3              <1>     inc     bl      ; next color index value
6009 00002CC7 EBBD              <1>     jmp     short g_s_s_1
6010                            <1> g_s_s_3:
6011                            <1>     ;mov    dx, 3DAh ; VGAREG_ACTL_RESET
6012                            <1>     ; 03/08/2022
6013 00002CC9 B2DA              <1>     mov     dl, 0DAh
6014 00002CCB EC                <1>     in      al, dx
6015 00002CCC B020              <1>     mov     al, 20h
6016                            <1>     ;mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
6017                            <1>     ; 03/08/2022
6018 00002CCE B2C0              <1>     mov     dl, 0C0h
6019 00002CD0 EE                <1>     out     dx, al ; 20h -> set bit 5
6020                            <1>                    ; (after loading palette regs)
6021 00002CD1 C3                <1>     retn
6022                            <1>
6023                            <1> vga_write_char_attr:
6024                            <1> vga_write_char_only:
6025                            <1>     ; 08/07/2016 (TRDOS 386 = TRDOS v2.0)
6026                            <1>     ;
6027                            <1>     ; derived from 'Plex86/Bochs VGABios' source code
6028                            <1>     ; vgabios-0.7a (2011)
6029                            <1>     ; by the LGPL VGABios developers Team (2001-2008)
6030                            <1>     ; 'vgabios.c', 'biosfn_write_char_attr'
6031                            <1>     ; 'biosfn_write_char_only'
6032                            <1>
6033                            <1>     ; INPUT ->
6034                            <1>     ; [CRT_MODE] = current video mode (>7)
6035                            <1>     ; CX = Count of characters to write
6036                            <1>     ; AL = Character to write
6037                            <1>     ; BL = Color of character
6038                            <1>     ; OUTPUT ->
6039                            <1>     ; Regen buffer updated
6040                            <1>
6041 00002CD2 8A25[1E680000]    <1>     mov     ah, [CRT_MODE]
6042 00002CD8 668B15[9E760100]  <1>     mov     dx, [CURSOR_POSN] ; cursor pos for page 0
6043                            <1>
6044 00002CDF BE[3A680000]      <1>     mov     esi, vga_modes
6045 00002CE4 89F7              <1>     mov     edi, esi
6046 00002CE6 83C710            <1>     add     edi, vga_mode_count
6047                            <1> vga_wca_0:
6048 00002CE9 AC                <1>     lodsb
6049 00002CEA 38E0              <1>     cmp     al, ah ; [CRT_MODE]
6050 00002CEC 7405              <1>     je      short vga_wca_2
6051 00002CEE 39FE              <1>     cmp     esi, edi
6052 00002CF0 72F7              <1>     jb      short vga_wca_0
6053                            <1> vga_wca_1:
6054 00002CF2 C3                <1>     retn    ; nothing to do
6055                            <1> vga_wca_2:
6056 00002CF3 83C64F            <1>     add     esi, vga_memmodel - (vga_modes + 1)
6057                            <1>     ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
6058                            <1>
6059                            <1>     ; biosfn_write_char_attr (car,page,attr,count)
6060                            <1>     ; AL = car, page = 0, BL = attr, CX = count
6061 00002CF6 803E04            <1>     cmp     byte [esi], PLANAR4
6062 00002CF9 741D              <1>     je      short vga_wca_planar
6063 00002CFB 803E03            <1>     cmp     byte [esi], PLANAR1
6064 00002CFE 7418              <1>     je      short vga_wca_planar
6065                            <1> vga_wca_linear8:
6066                            <1>     ; while((count-->0) && (xcurs<nbcols))
6067                            <1>     ; CX = count
6068 00002D00 6621C9            <1>     and     cx, cx
6069 00002D03 74ED              <1>     jz      short vga_wca_1
6070 00002D05 3A15[20680000]    <1>     cmp     dl, [CRT_COLS]
6071 00002D0B 73E5              <1>     jnb     short vga_wca_1
6072                            <1>     ; write_gfx_char_lin(car,attr,xcurs,ycurs,nbcols);
6073                            <1>     ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
6074                            <1>     ; [CRT_COLS] = nbcols
6075 00002D0D E81E000000        <1>     call    write_gfx_char_lin
6076 00002D12 6649              <1>     dec     cx ; count
6077 00002D14 FEC2              <1>     inc     dl ; xcurs
6078 00002D16 EBE8              <1>     jmp     short vga_wca_linear8
6079                            <1> vga_wca_planar:
6080                            <1>     ; while((count-->0) && (xcurs<nbcols))
6081                            <1>     ; CX = count
6082 00002D18 6621C9            <1>     and     cx, cx
6083 00002D1B 74D5              <1>     jz      short vga_wca_1
6084 00002D1D 3A15[20680000]    <1>     cmp     dl, [CRT_COLS]
6085 00002D23 73CD              <1>     jnb     short vga_wca_1
6086                            <1>     ; write_gfx_char_pl4(car,attr,xcurs,ycurs,nbcols,cheight);
6087                            <1>     ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
6088                            <1>     ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
6089 00002D25 E8A7000000          <1>     call    write_gfx_char_pl4
6090 00002D2A 6649              <1>     dec     cx ; count
6091 00002D2C FEC2              <1>     inc     dl ; xcurs
6092 00002D2E EBE8              <1>     jmp     short vga_wca_planar
6093                            <1>
6094                            <1> write_gfx_char_lin:
6095                            <1>     ; 02/08/2022 (TRDOS 386 v2.0.5)
6096                            <1>     ; 08/01/2021
6097                            <1>     ; 05/01/2021 (TRDOS 386 v2.0.3)
6098                            <1>     ; 08/08/2016
6099                            <1>     ; 31/07/2016
6100                            <1>     ; 08/07/2016 (TRDOS 386 = TRDOS v2.0)
6101                            <1>     ;
6102                            <1>     ; derived from 'Plex86/Bochs VGABios' source code
6103                            <1>     ; vgabios-0.7a (2011)
6104                            <1>     ; by the LGPL VGABios developers Team (2001-2008)
6105                            <1>     ; 'vgabios.c', 'write_gfx_char_lin'
6106                            <1>
6107                            <1>     ; write_gfx_char_lin(car,attr,xcurs,ycurs,nbcols)
6108                            <1>     ; INPUT ->
6109                            <1>     ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
6110                            <1>     ; [CRT_COLS] = nbcols
6111                            <1>     ; OUTPUT ->
6112                            <1>     ; Regen buffer updated
6113                            <1>
6114 00002D30 51                <1>     push    ecx
6115 00002D31 53                <1>     push    ebx
6116 00002D32 52                <1>     push    edx
6117 00002D33 50                <1>     push    eax
6118                            <1>     ; addr=xcurs*8+ycurs*nbcols*64;
6119                            <1>     ; 08/08/2016
6120 00002D34 0FB6F0            <1>     movzx   esi, al ; car
6121                            <1>     ; 08/01/2021
6122                            <1>     ;movzx  eax, dh ; ycurs
6123                            <1>     ;mov    ah, [CRT_COLS] ; nbcols
6124                            <1>     ;mul    ah
6125 00002D37 A0[20680000]      <1>     mov     al, [CRT_COLS]
```

```
6126 00002D3C F6E6                <1>     mul     dh
6127                              <1>     ;shl    ax, 6 ; * 64
6128                              <1>     ;shl    ax, 3 ; 8 * ycurs * [CRT_COLS]
6129                              <1>     ; 02/08/2022
6130 00002D3E C1E003              <1>     shl     eax, 3
6131                              <1>     ;sub    dh, dh
6132                              <1>     ;shl    dx, 3 ; xcurs * 8
6133                              <1>     ;movzx  edi, dx
6134 00002D41 BF00000A00          <1>     mov     edi, 0A0000h
6135 00002D46 30F6                <1>     xor     dh, dh
6136 00002D48 6689D7              <1>     mov     di, dx
6137                              <1>     ;movzx  edi, dl
6138 00002D4B 66C1E703            <1>     shl     di, 3 ; xcurs * 8
6139                              <1>     ;xor    dh, dh
6140 00002D4F 8A15[22680000]      <1>     mov     dl, [CHAR_HEIGHT]
6141 00002D55 66F7E2              <1>     mul     dx
6142                              <1>     ; eax = ycurs*nbcols*8*[CHAR_HEIGHT]
6143                              <1>     ;add    edi, eax ; addr
6144                              <1>     ;add    edi, 0A0000h
6145 00002D58 6601C7              <1>     add     di, ax
6146                              <1>     ;shl    si, 3 ; car * 8
6147 00002D5B 30E4                <1>     xor     ah, ah
6148 00002D5D A0[22680000]        <1>     mov     al, [CHAR_HEIGHT]
6149 00002D62 66F7E6              <1>     mul     si
6150 00002D65 6689C6              <1>     mov     si, ax
6151                              <1>     ;; esi = src = car * 8
6152                              <1>     ; esi = src = car * [CHAR_HEIGHT]
6153                              <1>     ; i = 0
6154                              <1>     ;add    esi, vgafont8 ; fdata [src+i]
6155                              <1>     ; 08/08/2016
6156 00002D68 A1[2A830100]        <1>     mov     eax, [VGA_INT43H]
6157 00002D6D 09C0                <1>     or      eax, eax ; 0 ?
6158 00002D6F 743E                <1>     jz      short wfxl_7 ; yes, default font
6159                              <1>     ;cmp    eax, vgafont16
6160                              <1>         ;je short wgfxl_0
6161                              <1>     ;cmp    eax, vgafont14
6162                              <1>     ;je     short wgfxl_0
6163                              <1>     ;cmp    eax, vgafont8
6164                              <1>     ;je     short wgfxl_0
6165                              <1>     ;; 05/01/2021 (TRDOS 386 v2.0.3)
6166                              <1>     ;; user font
6167                              <1>     ;mov    eax, VGAFONTUSR ; 8x16 or 8x8 or 8x14 font
6168                              <1>     ;                       ; (256 characters)
6169                              <1> wgfxl_0:
6170 00002D71 01C6                <1>     add     esi, eax
6171                              <1> wgfxl_1:
6172 00002D73 28FF                <1>     sub     bh, bh ; i = 0
6173                              <1> wgfxl_2:
6174                              <1>     ; for(i=0;i<8;i++)
6175 00002D75 57                  <1>     push    edi ; addr
6176 00002D76 0FB605[20680000]    <1>     movzx   eax, byte [CRT_COLS] ; nbcols
6177 00002D7D F6E7                <1>     mul     bh ; nbcols*i
6178                              <1>     ;shl    ax, 3 ; i*nbcols*8
6179                              <1>     ; 02/08/2022
6180 00002D7F C1E003              <1>     shl     eax, 3
6181                              <1>     ; dest=addr+i*nbcols*8;
6182 00002D82 01C7                <1>     add     edi, eax ; dest + j ; j = 0
6183 00002D84 B180                <1>     mov     cl, 80h ; mask = 0x80;
6184                              <1>     ; esi = fdata + src + i
6185                              <1>     ; for(j=0;j<8;j++)
6186 00002D86 29D2                <1>     sub     edx, edx ; j = 0
6187                              <1> wgfxl_3:
6188 00002D88 8A06                <1>     mov     al, [esi] ; al = fdata[src+i]
6189 00002D8A 20C8                <1>     and     al, cl ; if (fdata[src+i] & mask)
6190 00002D8C 7402                <1>     jz      short wgfxl_4  ; data = 0, zf = 1
6191 00002D8E 88D8                <1>     mov     al, bl ; data = attr;
6192                              <1> wgfxl_4:
6193                              <1>     ; write_byte(0xa000,dest+j,data);
6194 00002D90 AA                  <1>     stosb  ; dest + j (+ 0A0000h)
6195                              <1>     ;inc    dl ; j++
6196                              <1>     ;cmp    dl, 8
6197 00002D91 80FA07              <1>     cmp     dl, 7
6198 00002D94 720E                <1>     jb      short wgfxl_5
6199 00002D96 5F                  <1>     pop     edi
6200                              <1>     ; 08/08/2016
6201                              <1>     ;cmp    bh, 7
6202                              <1>     ;jnb    short wgfxl_6
6203 00002D97 FEC7                <1>     inc     bh ; i++
6204 00002D99 3A3D[22680000]      <1>     cmp     bh, [CHAR_HEIGHT]
6205 00002D9F 7309                <1>     jnb     short wgfxl_6
6206 00002DA1 46                  <1>     inc     esi
6207 00002DA2 EBD1                <1>     jmp     short wgfxl_2
6208                              <1> wgfxl_5:
6209 00002DA4 D0E9                <1>     shr     cl, 1 ; mask >>= 1;
6210 00002DA6 FEC2                <1>     inc     dl ; j++
6211 00002DA8 EBDE                <1>         jmp     short wgfxl_3
6212                              <1> wgfxl_6:
6213 00002DAA 58                  <1>     pop     eax
6214 00002DAB 5A                  <1>     pop     edx
6215 00002DAC 5B                  <1>     pop     ebx
6216 00002DAD 59                  <1>     pop     ecx
6217 00002DAE C3                  <1>     retn
6218                              <1> wfxl_7:
6219                              <1>     ; 08/01/2021
6220                              <1>     ; 05/01/2021
6221                              <1>     ; Default font (8x8 or 8x14 or 8x16)
6222 00002DAF A0[22680000]        <1>     mov     al, [CHAR_HEIGHT]
6223 00002DB4 3C08                <1>     cmp     al, 8
6224 00002DB6 7507                <1>     jne     short wfxl_8
6225 00002DB8 B8[104D0100]        <1>     mov     eax, vgafont8
6226 00002DBD EBB2                <1>     jmp     short wgfxl_0
6227                              <1> wfxl_8:
6228 00002DBF 3C0E                <1>     cmp     al, 14
6229 00002DC1 7507                <1>     jne     short wfxl_9
6230 00002DC3 B8[10550100]        <1>     mov     eax, vgafont14
6231 00002DC8 EBA7                <1>     jmp     short wgfxl_0
6232                              <1> wfxl_9:
6233 00002DCA B8[10630100]        <1>     mov     eax, vgafont16
6234 00002DCF EBA0                <1>     jmp     short wgfxl_0
6235                              <1>
6236                              <1> write_gfx_char_pl4:
6237                              <1>     ; 02/08/2022 (TRDOS 386 Kernel v2.0.5)
6238                              <1>     ; 08/08/2016
6239                              <1>     ; 08/07/2016 (TRDOS 386 = TRDOS v2.0)
6240                              <1>     ;
6241                              <1>     ; derived from 'Plex86/Bochs VGABios' source code
6242                              <1>     ; vgabios-0.7a (2011)
6243                              <1>     ; by the LGPL VGABios developers Team (2001-2008)
6244                              <1>     ; 'vgabios.c', 'write_gfx_char_pl4'
6245                              <1>
6246                              <1>     ; write_gfx_char_pl4(car,attr,xcurs,ycurs,nbcols,cheight)
6247                              <1>     ; INPUT ->
6248                              <1>     ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
6249                              <1>     ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
```

```
                                <1>     ; OUTPUT ->
                                <1>     ; Regen buffer updated
                                <1>
00002DD1 51                     <1>     push    ecx
00002DD2 53                     <1>     push    ebx
00002DD3 52                     <1>     push    edx
00002DD4 50                     <1>     push    eax
                                <1> wgfxpl_f0:
                                <1>     ; switch(cheight)
00002DD5 8A25[22680000]         <1>     mov     ah, [CHAR_HEIGHT]
00002DDB 80FC10                 <1>     cmp     ah, 16 ; case 16:
00002DDE 7507                   <1>     jne     short wgfxpl_f1
                                <1>     ; fdata = &vgafont16;
00002DE0 BE[10630100]           <1>     mov     esi, vgafont16
00002DE5 EB13                   <1>     jmp     short wgfxpl_f3
                                <1> wgfxpl_f1:
00002DE7 80FC0E                 <1>     cmp     ah, 14 ; case 14:
00002DEA 7507                   <1>     jne     short wgfxpl_f2
00002DEC BE[10550100]           <1>     mov     esi, vgafont14
00002DF1 EB07                   <1>     jmp     short wgfxpl_f3
                                <1> wgfxpl_f2:
                                <1>     ; default:
                                <1>     ;  fdata = &vgafont8;
00002DF3 BE[104D0100]           <1>     mov     esi, vgafont8
00002DF8 B408                   <1>     mov     ah, 8
                                <1> wgfxpl_f3:
                                <1>     ; al = car
00002DFA F6E4                   <1>     mul     ah ; ah = cheight
00002DFC 25FFFF0000             <1>     and     eax, 0FFFFh ; car * cheight
                                <1>     ; src = car * cheight;
00002E01 01C6                   <1>     add     esi, eax ; esi = fdata[src+i]
                                <1>     ; addr=xcurs*8+ycurs*nbcols*64;
00002E03 88F0                   <1>     mov     al, dh ; ycurs
00002E05 8A25[20680000]         <1>     mov     ah, [CRT_COLS] ; nbcols
00002E0B F6E4                   <1>     mul     ah
                                <1>     ; 08/08/2016
                                <1>     ;shl    ax, 6 ; * 64
                                <1>     ;shl    ax, 3 ; * 8
                                <1>     ; 02/08/2022
00002E0D C1E003                 <1>     shl     eax, 3
                                <1>     ;sub    dh, dh ; 0
                                <1>     ;shl    dx, 3 ; xcurs * 8
                                <1>     ;movzx  edi, dx
00002E10 0FB6FA                 <1>     movzx   edi, dl
                                <1>     ;shl    di, 3 ; xcurs * 8
                                <1>     ; 02/08/2022
00002E13 C1E703                 <1>     shl     edi, 3
00002E16 30F6                   <1>     xor     dh, dh
00002E18 8A15[22680000]         <1>     mov     dl, [CHAR_HEIGHT]
00002E1E 66F7E2                 <1>     mul     dx
                                <1>     ; eax = ycurs*nbcols*8*[CHAR_HEIGHT]
00002E21 01C7                   <1>     add     edi, eax ; addr
00002E23 81C700000A00           <1>     add     edi, 0A0000h
                                <1>     ;
                                <1>     ; outw(VGAREG_SEQU_ADDRESS, 0x0f02);
                                <1>     ; outw(VGAREG_GRDC_ADDRESS, 0x0205);
00002E29 66BAC403               <1>     mov     dx, 3C4h ; VGAREG_SEQU_ADDRESS
00002E2D 66B8020F               <1>     mov     ax, 0F02h
00002E31 66EF                   <1>     out     dx, ax
00002E33 66BACE03               <1>     mov     dx, 3CEh ; VGAREG_GRDC_ADDRESS
00002E37 66B80502               <1>     mov     ax, 0205h
00002E3B 66EF                   <1>     out     dx, ax
                                <1>     ;
00002E3D 66BACE03               <1>     mov     dx, 3CEh ; VGAREG_GRDC_ADDRESS
00002E41 F6C380                 <1>     test    bl, 80h ; if(attr&0x80)
00002E44 7406                   <1>     jz      short wgfxpl_f4 ; else
                                <1>     ; outw(VGAREG_GRDC_ADDRESS, 0x1803);
00002E46 66B80318               <1>     mov     ax, 1803h
00002E4A EB04                   <1>     jmp     short wgfxpl_f5
                                <1> wgfxpl_f4:
                                <1>     ; outw(VGAREG_GRDC_ADDRESS, 0x0003);
00002E4C 66B80300               <1>     mov     ax, 0003h
                                <1> wgfxpl_f5:
00002E50 66EF                   <1>     out     dx, ax
                                <1>     ;
00002E52 28FF                   <1>     sub     bh, bh ; i = 0
                                <1> wgfxpl_0:
                                <1>     ; for(i=0;i<cheight;i++)
00002E54 57                     <1>     push    edi ; addr
00002E55 0FB605[20680000]       <1>     movzx   eax, byte [CRT_COLS] ; nbcols
00002E5C F6E7                   <1>     mul     bh ; nbcols*i
                                <1>     ; dest=addr+i*nbcols
00002E5E 01C7                   <1>     add     edi, eax ; dest
00002E60 B580                   <1>     mov     ch, 80h ; mask = 0x80;
                                <1>     ; for(j=0;j<8;j++)
00002E62 28C9                   <1>     sub     cl, cl ; j = 0
                                <1> wgfxpl_1:
00002E64 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
00002E66 88EC                   <1>     mov     ah, ch
00002E68 B008                   <1>     mov     al, 8
00002E6A 66EF                   <1>     out     dx, ax
00002E6C 8A07                   <1>     mov     al, [edi] ; ? (io delay?)
                                <1>     ;
00002E6E 28C0                   <1>     sub     al, al ; attr = 0
                                <1>     ; if (fdata[src+i] & mask)
00002E70 842E                   <1>     test    byte [esi], ch
00002E72 7404                   <1>     jz      short wgfxpl_2  ; zf = 1
                                <1>     ; write_byte(0xa000,dest,attr&0x0f);
00002E74 88D8                   <1>     mov     al, bl ; attr;
00002E76 240F                   <1>     and     al, 0Fh ; attr&0x0f
                                <1> wgfxpl_2:
                                <1>     ; write_byte(0xa000,dest,0x00);
00002E78 8807                   <1>     mov     [edi], al ; dest (+ 0A0000h)
00002E7A FEC1                   <1>     inc     cl ; j++
00002E7C 80F908                 <1>     cmp     cl, 8
00002E7F 72E3                   <1>     jb      short wgfxpl_1
00002E81 5F                     <1>     pop     edi
                                <1>     ; 08/08/2016
                                <1>     ;cmp    bh, 7
                                <1>     ;jnb    short wgfxpl_3
00002E82 FEC7                   <1>     inc     bh ; i++
00002E84 3A3D[22680000]         <1>     cmp     bh, [CHAR_HEIGHT]
00002E8A 7303                   <1>     jnb     short wgfxpl_3
00002E8C 46                     <1>     inc     esi
00002E8D EBC5                   <1>     jmp     short wgfxpl_0
                                <1> wgfxpl_3:
                                <1>     ;mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
00002E8F 66B808FF               <1>     mov     ax, 0FF08h
00002E93 66EF                   <1>     out     dx, ax
00002E95 66B80500               <1>     mov     ax, 0005h
```

```
6374 00002E99 66EF              <1>     out     dx, ax
6375 00002E9B 66B80300          <1>     mov     ax, 0003h
6376 00002E9F 66EF              <1>     out     dx, ax
6377                            <1>     ;
6378 00002EA1 58                <1>     pop     eax
6379 00002EA2 5A                <1>     pop     edx
6380 00002EA3 5B                <1>     pop     ebx
6381 00002EA4 59                <1>     pop     ecx
6382 00002EA5 C3                <1>     retn
6383                            <1>
6384                            <1> vga_write_pixel:
6385                            <1>     ; 02/08/2022 (TRDOS 386 Kerbel v2.0.5)
6386                            <1>     ; 09/07/2016 (TRDOS 386 = TRDOS v2.0)
6387                            <1>     ;
6388                            <1>     ; derived from 'Plex86/Bochs VGABios' source code
6389                            <1>     ; vgabios-0.7a (2011)
6390                            <1>     ; by the LGPL VGABios developers Team (2001-2008)
6391                            <1>     ; 'vgabios.c', 'biosfn_write_pixel'
6392                            <1>
6393                            <1>     ; INPUT ->
6394                            <1>     ;       DX = row (0-239)
6395                            <1>     ;       CX = column (0-799)
6396                            <1>     ;       AL = pixel value
6397                            <1>     ;       (AH = [CRT_MODE])
6398                            <1>     ; OUTPUT ->
6399                            <1>     ;       none
6400                            <1>
6401 00002EA6 88C3              <1>     mov     bl, al ; pixel value
6402                            <1>     ;mov    ah, [CRT_MODE]
6403 00002EA8 BE[3A680000]      <1>     mov     esi, vga_modes
6404 00002EAD 89F7              <1>     mov     edi, esi
6405 00002EAF 83C710            <1>     add     edi, vga_mode_count
6406                            <1> vga_wp_0:
6407 00002EB2 AC                <1>     lodsb
6408 00002EB3 38E0              <1>     cmp     al, ah ; [CRT_MODE]
6409 00002EB5 7405              <1>     je      short vga_wp_1
6410 00002EB7 39FE              <1>     cmp     esi, edi
6411 00002EB9 72F7              <1>     jb      short vga_wp_0
6412 00002EBB C3                <1>     retn    ; nothing to do
6413                            <1> vga_wp_1:
6414 00002EBC 83C64F            <1>     add     esi, vga_memmodel - (vga_modes + 1)
6415                            <1>     ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
6416 00002EBF BF00000A00        <1>     mov     edi, 0A0000h
6417                            <1>     ;
6418 00002EC4 803E04            <1>     cmp     byte [esi], PLANAR4
6419 00002EC7 741C              <1>     je      short vga_wp_planar
6420 00002EC9 803E03            <1>     cmp     byte [esi], PLANAR1
6421 00002ECC 7417              <1>     je      short vga_wp_planar
6422                            <1> vga_wp_linear8:
6423                            <1>     ; addr=CX+DX*(read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS)*8);
6424 00002ECE 0FB605[20680000]  <1>     movzx   eax, byte [CRT_COLS] ; = [VGA_COLS] ; nbcols
6425                            <1>             ;shl    ax, 3 ; * 8
6426                            <1>     ; 02/08/2022
6427 00002ED5 C1E003            <1>     shl     eax, 3
6428 00002ED8 66F7E2            <1>     mul     dx
6429 00002EDB 50                <1>     push    eax
6430                            <1>     ;mov    edi, 0A0000h
6431 00002EDC 6601CF            <1>     add     di, cx
6432 00002EDF 58                <1>     pop     eax
6433 00002EE0 01C7              <1>     add     edi, eax ; addr
6434                            <1>     ; write_byte(0xa000,addr,AL);
6435 00002EE2 881F              <1>     mov     [edi], bl
6436 00002EE4 C3                <1>     retn
6437                            <1> vga_wp_planar:
6438                            <1>     ; addr = CX/8+DX*read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS);
6439 00002EE5 0FB7C1            <1>     movzx   eax, cx
6440 00002EE8 66C1E803          <1>     shr     ax, 3 ; CX/8
6441 00002EEC 50                <1>     push    eax
6442 00002EED 28E4              <1>     sub     ah, ah ; 0
6443 00002EEF A0[20680000]      <1>     mov     al, [CRT_COLS] ; = [VGA_COLS] ; nbcols
6444 00002EF4 66F7E2            <1>     mul     dx
6445                            <1>     ;mov    edi, 0A0000h
6446 00002EF7 6601C7            <1>         add     di, ax
6447 00002EFA 58                <1>     pop     eax
6448 00002EFB 01C7              <1>     add     edi, eax ; addr
6449 00002EFD 80E107            <1>     and     cl, 7
6450 00002F00 B580              <1>     mov     ch, 80h ; mask
6451 00002F02 D2ED              <1>     shr     ch, cl  ; mask = 0x80 >> (CX & 0x07);
6452                            <1>
6453                            <1>     ; outw(VGAREG_GRDC_ADDRESS, (mask << 8) | 0x08);
6454 00002F04 66BACE03          <1>     mov     dx, 3CEh ; VGAREG_GRDC_ADDRESS
6455 00002F08 88EC              <1>     mov     ah, ch
6456 00002F0A B008              <1>     mov     al, 8
6457 00002F0C 66EF              <1>     out     dx, ax
6458                            <1>     ; outw(VGAREG_GRDC_ADDRESS, 0x0205);
6459 00002F0E 66B80502          <1>     mov     ax, 0205h
6460 00002F12 66EF              <1>     out     dx, ax
6461                            <1>     ; data = read_byte(0xa000,addr);
6462 00002F14 8A07              <1>     mov     al, [edi] ; (delay?)
6463                            <1>     ; if (AL & 0x80)
6464                            <1>     ; {
6465                            <1>     ;  outw(VGAREG_GRDC_ADDRESS, 0x1803);
6466                            <1>     ; }
6467 00002F16 F6C380            <1>     test    bl, 80h
6468 00002F19 7406              <1>     jz      short vga_wp_2
6469 00002F1B 66B80318          <1>     mov     ax, 1803h
6470 00002F1F 66EF              <1>     out     dx, ax
6471                            <1> vga_wp_2:
6472                            <1>     ; write_byte(0xa000,addr,AL);
6473 00002F21 881F              <1>     mov     [edi], bl
6474                            <1>     ;
6475                            <1>     ;mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
6476 00002F23 66B808FF          <1>     mov     ax, 0FF08h
6477 00002F27 66EF              <1>     out     dx, ax
6478 00002F29 66B80500          <1>     mov     ax, 0005h
6479 00002F2D 66EF              <1>     out     dx, ax
6480 00002F2F 66B80300          <1>     mov     ax, 0003h
6481 00002F33 66EF              <1>     out     dx, ax
6482                            <1>     ;
6483 00002F35 C3                <1>     retn
6484                            <1>
6485                            <1> vga_read_pixel:
6486                            <1>     ; 02/08/2022 (TRDOS 386 Kernel v2.0.5)
6487                            <1>     ; 09/07/2016 (TRDOS 386 = TRDOS v2.0)
6488                            <1>     ;
6489                            <1>     ; derived from 'Plex86/Bochs VGABios' source code
6490                            <1>     ; vgabios-0.7a (2011)
6491                            <1>     ; by the LGPL VGABios developers Team (2001-2008)
6492                            <1>     ; 'vgabios.c', 'biosfn_read_pixel'
6493                            <1>
6494                            <1>     ; INPUT ->
6495                            <1>     ;       DX = row (0-239)
6496                            <1>     ;       CX = column (0-799)
6497                            <1>     ;       (AH = [CRT_MODE])
```

```
6498                                    <1>     ; OUTPUT ->
6499                                    <1>     ;       AL = pixel value
6500                                    <1>
6501                                    <1>     ;mov    ah, [CRT_MODE]
6502 00002F36 BE[3A680000]             <1>     mov     esi, vga_modes
6503 00002F3B 89F7                     <1>     mov     edi, esi
6504 00002F3D 83C710                   <1>     add     edi, vga_mode_count
6505                                    <1> vga_rp_0:
6506 00002F40 AC                       <1>     lodsb
6507 00002F41 38E0                     <1>     cmp     al, ah ; [CRT_MODE]
6508 00002F43 7405                     <1>     je      short vga_rp_1
6509 00002F45 39FE                     <1>     cmp     esi, edi
6510 00002F47 72F7                     <1>     jb      short vga_rp_0
6511 00002F49 C3                       <1>     retn    ; nothing to do
6512                                    <1> vga_rp_1:
6513 00002F4A 83C64F                   <1>     add     esi, vga_memmodel - (vga_modes + 1)
6514                                    <1>     ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
6515 00002F4D BF00000A00               <1>     mov     edi, 0A0000h
6516                                    <1>     ;
6517 00002F52 803E04                   <1>     cmp     byte [esi], PLANAR4
6518 00002F55 741C                     <1>     je      short vga_rp_planar
6519 00002F57 803E03                   <1>     cmp     byte [esi], PLANAR1
6520 00002F5A 7417                     <1>     je      short vga_rp_planar
6521                                    <1> vga_rp_linear8:
6522                                    <1>     ; addr=CX+DX*(read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS)*8);
6523 00002F5C 0FB605[20680000]         <1>     movzx   eax, byte [CRT_COLS] ; = [VGA_COLS] ; nbcols
6524                                    <1>             ;shl    ax, 3 ; * 8
6525                                    <1>     ; 02/08/2022
6526 00002F63 C1E003                   <1>     shl     eax, 3
6527 00002F66 66F7E2                   <1>     mul     dx
6528 00002F69 50                       <1>     push    eax
6529                                    <1>     ;mov    edi, 0A0000h
6530 00002F6A 6601CF                   <1>     add     di, cx
6531 00002F6D 58                       <1>     pop     eax
6532 00002F6E 01C7                     <1>     add     edi, eax ; addr
6533                                    <1>     ; attr=read_byte(0xa000,addr);
6534 00002F70 8A07                     <1>     mov     al, [edi] ; pixel value
6535 00002F72 C3                       <1>     retn
6536                                    <1> vga_rp_planar:
6537                                    <1>     ; addr = CX/8+DX*read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS);
6538 00002F73 0FB7C1                   <1>     movzx   eax, cx
6539 00002F76 66C1E803                 <1>     shr     ax, 3 ; CX/8
6540 00002F7A 50                       <1>     push    eax
6541 00002F7B 28E4                     <1>     sub     ah, ah ; 0
6542 00002F7D A0[20680000]             <1>     mov     al, [CRT_COLS] ; = [VGA_COLS] ; nbcols
6543 00002F82 66F7E2                   <1>     mul     dx
6544                                    <1>     ;mov    edi, 0A0000h
6545 00002F85 6601C7                   <1>         add     di, ax
6546 00002F88 58                       <1>     pop     eax
6547 00002F89 01C7                     <1>     add     edi, eax ; addr
6548 00002F8B 80E107                   <1>     and     cl, 7
6549 00002F8E B580                     <1>     mov     ch, 80h ; mask
6550 00002F90 D2ED                     <1>     shr     ch, cl  ; mask = 0x80 >> (CX & 0x07);
6551                                    <1>     ; attr = 0x00;
6552 00002F92 30DB                     <1>     xor     bl, bl ; attr = bl = 0,
6553 00002F94 30C9                     <1>     xor     cl, cl ; i = cl = 0
6554                                    <1>     ; for(i=0;i<4;i++)
6555                                    <1>             ; {
6556                                    <1>             ;  outw(VGAREG_GRDC_ADDRESS, (i << 8) | 0x04);
6557                                    <1>             ;  data = read_byte(0xa000,addr) & mask;
6558                                    <1>             ;  if (data > 0) attr |= (0x01 << i);
6559                                    <1>             ; }
6560                                    <1> vga_rp_2:
6561 00002F96 88CC                     <1>     mov     ah, cl ; i << 8
6562 00002F98 B004                     <1>     mov     al, 4  ; | 0x04
6563 00002F9A 66BACE03                 <1>     mov     dx, 3CEh ; VGAREG_GRDC_ADDRESS
6564 00002F9E 66EF                     <1>     out     dx, ax
6565                                    <1>     ; data = read_byte(0xa000,addr) & mask;
6566 00002FA0 8A07                     <1>     mov     al, [edi]
6567 00002FA2 20E8                     <1>     and     al, ch ; & mask
6568                                    <1>     ; if (data > 0) attr |= (0x01 << i);
6569 00002FA4 08C0                     <1>     or      al, al
6570 00002FA6 7408                     <1>     jz      short vga_rp_3 ; al = 0
6571 00002FA8 B701                     <1>     mov     bh, 1
6572 00002FAA D2E7                     <1>     shl     bh, cl ; (0x01 << i)
6573 00002FAC 08FB                     <1>     or      bl, bh ; attr |= (0x01 << i)
6574 00002FAE 88D8                     <1>     mov     al, bl ; pixel value
6575                                    <1> vga_rp_3:
6576 00002FB0 C3                       <1>     retn
6577                                    <1>
6578                                    <1> vga_beeper:
6579                                    <1>     ; 04/08/2016  (TRDOS 386 = TRDOS v2.0)
6580 00002FB1 FB                       <1>     sti
6581                                    <1>     ;mov    bh, [ACTIVE_PAGE]
6582 00002FB2 E917F4FFFF               <1>         jmp     beeper_gfx
6583                                    <1>
6584                                    <1> vga_write_teletype:
6585                                    <1>     ; 03/08/2022 (TRDOS 386 Kernel v2.0.5)
6586                                    <1>     ; 12/04/2021 (TRDOS 386 v2.0.3, 32 bit push/pop)
6587                                    <1>     ; 09/12/2017
6588                                    <1>     ; 06/08/2016
6589                                    <1>     ; 04/08/2016
6590                                    <1>     ; 01/08/2016
6591                                    <1>     ; 31/07/2016
6592                                    <1>     ; 09/07/2016 (TRDOS 386 = TRDOS v2.0)
6593                                    <1>     ;
6594                                    <1>     ; derived from 'Plex86/Bochs VGABios' source code
6595                                    <1>     ; vgabios-0.7a (2011)
6596                                    <1>     ; by the LGPL VGABios developers Team (2001-2008)
6597                                    <1>     ; 'vgabios.c', 'biosfn_write_teletype'
6598                                    <1>     ; 'biosfn_write_char_only'
6599                                    <1>
6600                                    <1>     ; INPUT ->
6601                                    <1>     ; [CRT_MODE] = current video mode (>7)
6602                                    <1>     ; AL = Character to write
6603                                    <1>     ; BL = Color of character
6604                                    <1>     ; OUTPUT ->
6605                                    <1>     ; Regen buffer updated
6606                                    <1>
6607                                    <1>     ; biosfn_write_teletype (car, page, attr, flag)
6608                                    <1>     ; car = character (AL)
6609                                    <1>     ; page = 0
6610                                    <1>     ; attr = color (BL)
6611                                    <1>     ; 'flag' not used
6612                                    <1>
6613 00002FB7 8A25[1E680000]           <1>     mov     ah, [CRT_MODE]
6614 00002FBD 88C7                     <1>     mov     bh, al ; character
6615 00002FBF 668B15[9E760100]         <1>     mov     dx, [CURSOR_POSN] ; cursor pos for page 0
6616                                    <1>
6617 00002FC6 BE[42680000]             <1>     mov     esi, vga_g_modes
6618 00002FCB 89F7                     <1>     mov     edi, esi
6619 00002FCD 83C708                   <1>     add     edi, vga_g_mode_count
6620                                    <1> vga_wtty_0:
6621 00002FD0 AC                       <1>     lodsb
```

```
6622 00002FD1 38E0              <1>     cmp     al, ah ; [CRT_MODE]
6623 00002FD3 7405              <1>     je      short vga_wtty_2
6624 00002FD5 39FE              <1>     cmp     esi, edi
6625 00002FD7 72F7              <1>     jb      short vga_wtty_0
6626                            <1> vga_wtty_1:
6627 00002FD9 C3                <1>     retn    ; nothing to do
6628                            <1> vga_wtty_2:
6629 00002FDA 80FF07            <1>     cmp     bh, 07h ; bell (beep)
6630 00002FDD 74D2              <1>     je      short vga_beeper  ; u11
6631 00002FDF 80FF08            <1>     cmp     bh, 08h ; backspace
6632 00002FE2 7508              <1>     jne     short vga_wtty_3
6633                            <1>     ; if(xcurs>0)xcurs--;
6634 00002FE4 08D2              <1>     or      dl, dl ; xcurs (column)
6635 00002FE6 74F1              <1>     jz      short vga_wtty_1
6636 00002FE8 FECA              <1>     dec     dl ; xcurs--;
6637 00002FEA EB55              <1>         jmp     short vga_wtty_12
6638                            <1> vga_wtty_3:
6639 00002FEC 80FF0D            <1>     cmp     bh, 0Dh ; carriage return (\r)
6640 00002FEF 7504              <1>     jne     short vga_wtty_4
6641                            <1>     ; xcurs=0;
6642 00002FF1 28D2              <1>     sub     dl, dl ; 0
6643 00002FF3 EB4C              <1>         jmp     short vga_wtty_12
6644                            <1> vga_wtty_4:
6645 00002FF5 80FF0A            <1>     cmp     bh, 0Ah ; new line (\n)
6646 00002FF8 7504              <1>     jne     short vga_wtty_5
6647                            <1>     ; ycurs++;
6648 00002FFA FEC6              <1>     inc     dh ; next row
6649 00002FFC EB5E              <1>         jmp     short vga_wtty_11
6650                            <1> vga_wtty_5:
6651 00002FFE 80FF09            <1>     cmp     bh, 09h ; tab stop
6652 00003001 7523              <1>     jne     short vga_wtty_8
6653 00003003 88D0              <1>     mov     al, dl
6654                            <1>     ;cbw
6655 00003005 30E4              <1>     xor     ah, ah ; 09/12/2017
6656 00003007 B108              <1>     mov     cl, 8
6657 00003009 F6F1              <1>     div     cl
6658 0000300B 28E1              <1>     sub     cl, ah
6659                            <1>     ;
6660 0000300D B720              <1>     mov     bh, 20h ; space
6661                            <1> vga_wtty_6: ; tab stop loop
6662                            <1>     ;push   cx
6663                            <1>     ;push   bx
6664                            <1>     ; 12/04/2021
6665 0000300F 51                <1>     push    ecx
6666 00003010 53                <1>     push    ebx
6667 00003011 E810000000        <1>     call    vga_wtty_8
6668                            <1>     ;pop    bx  ; bh = character, bl = color
6669                            <1>     ;pop    cx
6670                            <1>     ; 12/04/2021
6671 00003016 5B                <1>     pop     ebx  ; bh = character, bl = color
6672 00003017 59                <1>     pop     ecx
6673 00003018 FEC9              <1>     dec     cl
6674 0000301A 7409              <1>     jz      short vga_wtty_7
6675 0000301C 668B15[9E760100]  <1>     mov     dx, [CURSOR_POSN] ; new cursor position (pg 0)
6676 00003023 EBEA              <1>     jmp     short vga_wtty_6
6677                            <1> vga_wtty_7:
6678 00003025 C3                <1>     retn
6679                            <1>     ;
6680                            <1> vga_wtty_8:
6681 00003026 83C64F            <1>     add     esi, vga_g_memmodel - (vga_g_modes + 1)
6682                            <1>     ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
6683 00003029 BF00000A00        <1>     mov     edi, 0A0000h
6684                            <1>     ;
6685 0000302E 88F8              <1>     mov     al, bh ; character
6686                            <1>     ;
6687 00003030 803E04            <1>     cmp     byte [esi], PLANAR4
6688 00003033 7414              <1>     je      short vga_wtty_planar
6689 00003035 803E03            <1>     cmp     byte [esi], PLANAR1
6690 00003038 740F              <1>     je      short vga_wtty_planar
6691                            <1> vga_wtty_linear8:
6692                            <1>     ; write_gfx_char_lin(car,attr,xcurs,ycurs,nbcols);
6693                            <1>     ; AL = car, BL = attr (color), DL = xcurs, DH = ycurs,
6694                            <1>     ; [CRT_COLS] = nbcols
6695 0000303A E8F1FCFFFF        <1>     call    write_gfx_char_lin
6696 0000303F EB0D              <1>     jmp     short vga_wtty_9
6697                            <1>
6698                            <1> vga_wtty_12:
6699                            <1>     ; 09/07/2016
6700                            <1>     ; set cursor position
6701                            <1>     ; NOTE: Hardware cursor position will not be set
6702                            <1>     ;   in any VGA modes (>7)
6703                            <1>     ;   But, cursor position will be saved into
6704                            <1>     ;   [CURSOR_POSN].
6705                            <1>     ;   TRDOS 386 (TRDOS v2.0) uses only one page
6706                            <1>     ;   (page 0) for all graphics modes.
6707                            <1>
6708 00003041 668915[9E760100]  <1>     mov     [CURSOR_POSN], dx ; save cursor pos for pg 0
6709                            <1>     ; 04/08/2016
6710                            <1>     ;mov    bh, [ACTIVE_PAGE] ; = 0
6711                            <1>     ;call   _set_cpos
6712 00003048 C3                <1>     retn
6713                            <1>
6714                            <1> vga_wtty_planar:
6715                            <1>     ; write_gfx_char_pl4(car,attr,xcurs,ycurs,nbcols,cheight);
6716                            <1>     ; AL = car, BL = attr (color), DL = xcurs, DH = ycurs,
6717                            <1>     ; [CRT_COLS]= nbcols, [CHAR_HEIGHT] = cheight
6718 00003049 E883FDFFFF        <1>     call    write_gfx_char_pl4
6719                            <1> vga_wtty_9:
6720 0000304E FEC2              <1>     inc     dl ; xcurs++;
6721                            <1> vga_wtty_10:
6722                            <1>     ; Do we need to wrap ?
6723                            <1>     ; if(xcurs==nbcols)
6724 00003050 3A15[20680000]    <1>     cmp     dl, [CRT_COLS] ; [VGA_COLS]
6725 00003056 7204              <1>     jb      short vga_wtty_11 ; no
6726 00003058 28D2              <1>     sub     dl, dl ; xcurs=0;
6727 0000305A FEC6              <1>     inc     dh ;  ycurs++;
6728                            <1> vga_wtty_11:
6729                            <1>     ; Do we need to scroll ?
6730                            <1>     ; if(ycurs==nbrows)
6731 0000305C 3A35[26680000]    <1>     cmp     dh, [VGA_ROWS]
6732 00003062 72DD              <1>     jb      short vga_wtty_12 ; no
6733                            <1>     ;
6734                            <1>     ; biosfn_scroll (nblines,attr,rul,cul,rlr,clr,page,dir)
6735                            <1>     ; al = nblines = 1, bl = attr (color) = 0
6736                            <1>     ; ch = rul, cl = cul, dh = rlr, dl = clr, page = 0
6737                            <1>     ; dir = SCROLL_UP
6738                            <1>
6739 00003064 B001              <1>     mov     al, 1
6740 00003066 28DB              <1>     sub     bl, bl ; 0 ; blank/black line (attr=0) will be used
6741                            <1>     ;sub    cx, cx ; 0,0
6742                            <1>     ; 03/08/2022
6743 00003068 29C9              <1>     sub     ecx, ecx
6744                            <1>
6745                            <1>     ; 06/08/2016
```

```
6746 0000306A 8A35[26680000]      <1>     mov     dh, [VGA_ROWS]
6747 00003070 FECE                <1>     dec     dh ; nbrows -1
6748                              <1>
6749                              <1>     ;push   dx ; 04/08/2016
6750                              <1>     ; 12/04/2021
6751 00003072 52                  <1>     push    edx
6752 00003073 8A15[20680000]      <1>     mov     dl, [CRT_COLS]
6753 00003079 FECA                <1>     dec     dl ; nbcols -1
6754                              <1>
6755 0000307B 8A25[1E680000]      <1>     mov     ah, [CRT_MODE]
6756                              <1>
6757                              <1>     ; biosfn_scroll(0x01,0x00,0,0,nbrows-1,nbcols-1,page,SCROLL_UP);
6758 00003081 E843F5FFFF          <1>     call    vga_graphics_up
6759                              <1>     ; 04/08/2016
6760                              <1>     ;pop    dx
6761                              <1>     ; 12/04/2021
6762 00003086 5A                  <1>     pop     edx
6763                              <1>
6764                              <1>     ;dec    dh ; ycurs-=1
6765 00003087 EBB8                <1>     jmp     short vga_wtty_12
6766                              <1>
6767                              <1> font_setup:
6768                              <1>     ; 03/08/2022 (TRDOS 386 v2.0.5)
6769                              <1>     ; 09/01/2021 (TRDOS 386 v2.0.3)
6770                              <1>     ; 09/07/2016
6771                              <1>     ; character generator (font loading) functions
6772                              <1>     ;
6773                              <1>     ; derived from 'Plex86/Bochs VGABios' source code
6774                              <1>     ; vgabios-0.7a (2011)
6775                              <1>     ; by the LGPL VGABios developers Team (2001-2008)
6776                              <1>     ; 'vgabios.c', 'int10_func'
6777                              <1>
6778                              <1>     ; AX = 1100H ; Load User-Defined Font (EGA/VGA)
6779                              <1>     ;
6780                              <1>         ; BH =  height of each character (bytes per character definition)
6781                              <1>         ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
6782                              <1>     ; CX =  number of characters to redefine (<=256)
6783                              <1>         ; DX =  ASCII code of the first character defined at ES:BP
6784                              <1>         ; EBP =    address of font-definition information
6785                              <1>     ;       (in user's memory space)
6786                              <1>
6787                              <1>     ; case 0x11:
6788                              <1>             ; switch(GET_AL())
6789                              <1>             ; {
6790                              <1>     ; case 0x00:
6791                              <1>         ; case 0x10:
6792                              <1>         ; biosfn_load_text_user_pat(GET_AL(),ES,BP,CX,DX,GET_BL(),GET_BH());
6793                              <1>         ; break;
6794                              <1>
6795                              <1>     ; AX = 1110H ; Load and Activate User-Defined Font (EGA/VGA)
6796 00003089 08C0                <1>     or      al, al ; 0
6797 0000308B 7404                <1>     jz      short font_setup_0
6798 0000308D 3C10                <1>     cmp     al, 10h
6799 0000308F 7511                <1>     jne     short font_setup_1
6800                              <1> font_setup_0:
6801 00003091 E8CE000000          <1>     call    transfer_user_fonts
6802 00003096 721C                <1>     jc      short font_setup_error
6803 00003098 E8AD010000          <1>     call    load_text_user_pat
6804 0000309D E9D6EAFFFF          <1>         jmp     VIDEO_RETURN
6805                              <1> font_setup_1:
6806                              <1>     ; AX = 1101H ; Load ROM 8x14 Character Set (EGA/VGA)
6807                              <1>     ; case 0x01:
6808                              <1>         ; case 0x11:
6809                              <1>         ; biosfn_load_text_8_14_pat(GET_AL(),GET_BL());
6810                              <1>         ; break;
6811 000030A2 3C01                <1>     cmp     al, 1
6812 000030A4 7404                <1>     je      short font_setup_2
6813 000030A6 3C11                <1>     cmp     al, 11h
6814 000030A8 7511                <1>     jne     short font_setup_3
6815                              <1> font_setup_2:
6816                              <1>     ; AX = 1111H ; Load and Activate ROM 8x14 Character Set (EGA/VGA)
6817                              <1>     ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
6818 000030AA E8C9020000          <1>     call    load_text_8_14_pat
6819 000030AF E9C4EAFFFF          <1>         jmp     VIDEO_RETURN
6820                              <1> font_setup_error:
6821 000030B4 29C0                <1>     sub     eax, eax ; 0 -> fonts could not be loaded
6822 000030B6 E9C2EAFFFF          <1>     jmp     _video_return
6823                              <1> font_setup_3:
6824                              <1>     ; AX = 1102H ; Load ROM 8x8 Character Set (EGA/VGA)
6825                              <1>     ; case 0x02:
6826                              <1>         ; case 0x12:
6827                              <1>         ; biosfn_load_text_8_8_pat(GET_AL(),GET_BL());
6828                              <1>         ; break;
6829 000030BB 3C02                <1>     cmp     al, 2
6830 000030BD 7404                <1>     je      short font_setup_4
6831 000030BF 3C12                <1>     cmp     al, 12h
6832 000030C1 750A                <1>     jne     short font_setup_5
6833                              <1> font_setup_4:
6834                              <1>     ; AX = 1112H ; Load and Activate ROM 8x8 Character Set (EGA/VGA)
6835                              <1>     ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
6836 000030C3 E8E1020000          <1>     call    load_text_8_8_pat
6837 000030C8 E9ABEAFFFF          <1>         jmp     VIDEO_RETURN
6838                              <1> font_setup_5:
6839                              <1>     ; AX = 1104H ; Load ROM 8x16 Character Set (EGA/VGA)
6840                              <1>     ; case 0x04:
6841                              <1>         ; case 0x14:
6842                              <1>         ; biosfn_load_text_8_16_pat(GET_AL(),GET_BL());
6843                              <1>         ; break;
6844 000030CD 3C04                <1>     cmp     al, 4
6845 000030CF 7404                <1>     je      short font_setup_6
6846 000030D1 3C14                <1>     cmp     al, 14h
6847 000030D3 750A                <1>     jne     short font_setup_7
6848                              <1> font_setup_6:
6849                              <1>     ; AX = 1114H ; Load and Activate ROM 8x16 Character Set (EGA/VGA)
6850                              <1>     ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
6851 000030D5 E819030000          <1>     call    load_text_8_16_pat
6852 000030DA E999EAFFFF          <1>         jmp     VIDEO_RETURN
6853                              <1> font_setup_7:
6854                              <1>     ; Note: AX=1120h (Setup INT 1Fh, EXT_PTR) is not needed
6855                              <1>     ; for TRDOS 386 (TRDOS v2.0) video functionality;
6856                              <1>     ; because, originally EXT_PTR (font address) was used for
6857                              <1>     ; chars 80h to 0FFh (after the first 128 ASCII char fonts), for
6858                              <1>     ; CGA graphics mode; currenty, 'vgafont8' address has 256 chars!
6859                              <1>     ;
6860                              <1>     ; case 0x20:
6861                              <1>         ; biosfn_load_gfx_8_8_chars(ES,BP);
6862                              <1>         ; break;
6863                              <1>     ; case 0x21:
6864                              <1>         ; biosfn_load_gfx_user_chars(ES,BP,CX,GET_BL(),GET_DL());
6865                              <1>         ; break;
6866                              <1>     ; AX = 1121H ; Setup User-Defined Font for Graphics Mode (VGA)
6867                              <1>     ; BL   screen rows code: 00H = user-specified (in DL)
6868                              <1>         ;                        01H = 14 rows
6869                              <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>
000030DF 3C21                   <1>     cmp     al, 21h
000030E1 7531                   <1>     jne     short font_setup_9
                                <1>
                                <1>     ; TRDOS 386 modification !
                                <1>     ; dh = 0 -> 256 characters
                                <1>     ; dh = 80h -> second 128 characters
                                <1>     ; dh = 0FFh -> first 128 characters
                                <1>
                                <1>     ; 09/01/2021 (TRDOS 386 v2.0.3)
                                <1>     ;push   ebx
000030E3 51                     <1>     push    ecx
000030E4 52                     <1>     push    edx
000030E5 30D2                   <1>     xor     dl, dl
000030E7 88CF                   <1>     mov     bh, cl ; character height
                                <1>     ;mov    cx, 100h ; 256
                                <1>     ; 03/08/2022
000030E9 31C9                   <1>     xor     ecx, ecx
000030EB FEC5                   <1>     inc     ch
                                <1>     ; ecx = 100h
000030ED 08F6                   <1>     or      dh, dh ; 0
000030EF 7410                   <1>     jz      short  font_setup_8
000030F1 FECD                   <1>     dec     ch ; cx = 0
000030F3 80FEFF                 <1>     cmp     dh, 0FFh
000030F6 7409                   <1>     je      short font_setup_8 ; 1st 128 chars
                                <1>     ; 2nd 128 chars
000030F8 80FE80                 <1>     cmp     dh, 80h
000030FB 75B7                   <1>     jne     short font_setup_error ; invalid !
000030FD 88F1                   <1>     mov     cl, dh
000030FF 86F2                   <1>     xchg    dl, dh
                                <1>     ; number of chars, cx = 80h
                                <1>     ; start char, dl = 80h
                                <1> font_setup_8:
00003101 E85E000000             <1>     call    transfer_user_fonts
00003106 5A                     <1>     pop     edx
00003107 59                     <1>     pop     ecx
                                <1>     ;pop    ebx
00003108 72AA                   <1>     jc      short font_setup_error
                                <1>     ; ebp = user's font data address in system's memory space
0000310A E82F030000             <1>     call    load_gfx_user_chars
0000310F E964EAFFFF             <1>         jmp     VIDEO_RETURN
                                <1> font_setup_9:
                                <1>     ; case 0x22:
                                <1>         ; biosfn_load_gfx_8_14_chars(GET_BL());
                                <1>         ; break;
00003114 3C22                   <1>     cmp     al, 22h
00003116 750A                   <1>     jne     short font_setup_10
00003118 E85D030000             <1>     call    load_gfx_8_14_chars
0000311D E956EAFFFF             <1>         jmp     VIDEO_RETURN
                                <1> font_setup_10:
                                <1>     ; case 0x23:
                                <1>         ; biosfn_load_gfx_8_8_dd_chars(GET_BL());
                                <1>         ; break;
00003122 3C23                   <1>     cmp     al, 23h
00003124 750A                   <1>     jne     short font_setup_11
00003126 E890030000             <1>     call    load_gfx_8_8_chars
0000312B E948EAFFFF             <1>         jmp     VIDEO_RETURN
                                <1> font_setup_11:
                                <1>     ; case 0x24:
                                <1>         ; biosfn_load_gfx_8_16_chars(GET_BL());
                                <1>         ; break;
00003130 3C24                   <1>     cmp     al, 24h
00003132 750A                   <1>     jne     short font_setup_12
00003134 E8C3030000             <1>     call    load_gfx_8_16_chars
00003139 E93AEAFFFF             <1>         jmp     VIDEO_RETURN
                                <1> font_setup_12:
                                <1>     ; case 0x30:
                                <1>         ; biosfn_get_font_info(GET_BH(),&ES,&BP,&CX,&DX);
                                <1>         ; break;
0000313E 3C30                   <1>     cmp     al, 30h
00003140 750A                   <1>     jne     short font_setup_13
00003142 E8F6030000             <1>     call    get_font_info
                                <1>     ; eax = return value (info: 4 bytes for 4 parms)
                                <1>     ; eax = 0 -> invalid function (input)
00003147 E931EAFFFF             <1>         jmp     _video_return
                                <1> font_setup_13:
0000314C 3C03                   <1>     cmp     al, 03h ; AX = 1103h
0000314E 750D                   <1>     jne     short font_setup_14
                                <1>     ; biosfn_set_text_block_specifier:
                                <1>     ; BL = font block selector code
                                <1>     ; NOTE: TRDOS 386 only uses and sets font block 0
                                <1>     ; (It is as BL = 0 for TRDOS 386)
00003150 66BAC403               <1>     mov     dx, 3C4h ; VGAREG_SEQU_ADDRESS
                                <1>     ;mov    ah, bl
00003154 28E4                   <1>     sub     ah, ah ; 0
                                <1>     ;mov    al, 03h
00003156 66EF                   <1>     out     dx, ax
00003158 E91BEAFFFF             <1>     jmp      VIDEO_RETURN
                                <1>
                                <1> font_setup_14:
0000315D 29C0                   <1>     sub     eax, eax ; 0 = invalid function
0000315F E919EAFFFF             <1>         jmp     _video_return
                                <1>
                                <1> transfer_user_fonts:
                                <1>     ; 02/08/2022 (TRDOS 386 Kernel v2.0.5)
                                <1>     ; 19/01/2021
                                <1>     ; 09/01/2021
                                <1>     ; 05/01/2021 (TRDOS 386 v2.0.3)
                                <1>
                                <1>     ; BH =  height of each character (bytes per character)
                                <1>     ; CX =  number of characters to redefine (<=256)
                                <1>         ; DX =  ASCII code of the first character defined at EBP
                                <1>         ; EBP =    address of font-definition information
                                <1>     ;       (in user's memory space)
                                <1>
                                <1>     ; Modified registers: eax, edx, ecx, esi, edi, ebp
                                <1>     ;
                                <1>     ; output:
                                <1>     ;       ebp = user font data address in system memory
                                <1>
00003164 81E2FFFF0000           <1>     and     edx, 0FFFFh
0000316A 81E1FFFF0000           <1>     and     ecx, 0FFFFh
00003170 7537                   <1>     jnz     short transfer_user_fonts_5
                                <1>
00003172 09D2                   <1>     or      edx, edx
00003174 7531                   <1>     jnz     short transfer_user_fonts_4
00003176 09ED                   <1>     or      ebp, ebp
00003178 752D                   <1>     jnz     short transfer_user_fonts_4
                                <1>
                                <1>     ; cx = 0, dx = 0, ebp = 0
```

```
                                <1>     ; copy system font to user font
                                <1>
0000317A B140                   <1>     mov     cl, 64 ; 64 dwords
                                <1>
0000317C 80FF10                 <1>     cmp     bh, 16
0000317F 7417                   <1>     je      short transfer_user_fonts_2
00003181 80FF08                 <1>     cmp     bh, 8
00003184 7406                   <1>     je      short transfer_user_fonts_1
                                <1>
00003186 BD[10550100]           <1>     mov     ebp, vgafont14
0000318B C3                     <1>     retn
                                <1>
                                <1> transfer_user_fonts_1:
0000318C BF00500900             <1>     mov     edi, VGAFONT8USER
00003191 BE[104D0100]           <1>     mov     esi, vgafont8
00003196 EB0A                   <1>     jmp     short transfer_user_fonts_3
                                <1>
                                <1> transfer_user_fonts_2:
00003198 BF00400900             <1>     mov     edi, VGAFONT16USER
0000319D BE[10630100]           <1>     mov     esi, vgafont16
                                <1> transfer_user_fonts_3:
000031A2 89FD                   <1>     mov     ebp, edi
000031A4 F3A5                   <1>     rep     movsd
000031A6 C3                     <1>     retn
                                <1>
                                <1> transfer_user_fonts_4:
000031A7 F9                     <1>     stc
000031A8 C3                     <1>     retn
                                <1>
                                <1> transfer_user_fonts_5:
000031A9 09ED                   <1>     or      ebp, ebp
000031AB 74FA                   <1>     jz      short transfer_user_fonts_4 ; invalid address !
                                <1>
000031AD 6681F90001             <1>     cmp     cx, 256
000031B2 77F3                   <1>     ja      short transfer_user_fonts_4
000031B4 29D1                   <1>     sub     ecx, edx
000031B6 76EF                   <1>     jna     short transfer_user_fonts_4
                                <1>
000031B8 80FF0E                 <1>     cmp     bh, 14 ; 8x14 font
                                <1>                    ; (there is not an alternative buffer)
000031BB 7524                   <1>     jne     short transfer_user_fonts_6
                                <1>
                                <1>     ; use system's 8x14 font space if permission flag is 1
000031BD F605[4E9D0100]80       <1>     test    byte [ufont], 80h
000031C4 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>
                                <1>     ;push   edx
                                <1>     ; 02/08/2022
000031C6 87CA                   <1>     xchg    edx, ecx
                                <1>     ;xor    ah, ah
                                <1>     ; 02/08/2022
000031C8 31C0                   <1>     xor     eax, eax
000031CA 88F8                   <1>     mov     al, bh ; mov al, 14
                                <1>     ;mul    cx
                                <1>     ; 02/08/2022
000031CC F7E2                   <1>     mul     edx  ; char count * 14
                                <1>     ; 02/08/2022
000031CE 89CA                   <1>     mov     edx, ecx ; ascii code
000031D0 89C1                   <1>     mov     ecx, eax
                                <1>             ; ecx = byte count
                                <1>     ;pop    edx
                                <1>     ; 02/08/2022
                                <1>     ;xor    eax, eax
000031D2 30E4                   <1>     xor     ah, ah
000031D4 88F8                   <1>     mov     al, bh ; mov ax, 14 ; bytes per character
                                <1>     ;mul    dx
                                <1>     ;mov    dx, ax ; char offset
                                <1>     ; 02/08/2022
000031D6 F7E2                   <1>     mul     edx
000031D8 89C2                   <1>     mov     edx, eax ; char offset
000031DA BF[10550100]           <1>     mov     edi, vgafont14
000031DF EB48                   <1>     jmp     short transfer_user_fonts_8
                                <1> transfer_user_fonts_6:
000031E1 80FF08                 <1>     cmp     bh, 8 ; 8x8 font
000031E4 7520                   <1>     jne     short transfer_user_fonts_7 ; 8x16 font
                                <1>     ;shl    dx, 3 ; * 8
                                <1>     ;shl    cx, 3 ; * 8
                                <1>     ; 02/08/2022
000031E6 C1E203                 <1>     shl     edx, 3 ; byte offset
000031E9 C1E103                 <1>     shl     ecx, 3 ; byte count
                                <1>     ; 09/01/2021
000031EC BF00500900             <1>     mov     edi, VGAFONT8USER
000031F1 F605[4E9D0100]08       <1>     test    byte [ufont], 8  ; already loaded ?
000031F8 752F                   <1>     jnz     short transfer_user_fonts_8 ; yes
000031FA BE[104D0100]           <1>     mov     esi, vgafont8
000031FF E839000000             <1>     call    transfer_user_fonts_10
00003204 EB23                   <1>     jmp     short transfer_user_fonts_8
                                <1> transfer_user_fonts_7:
00003206 80FF10                 <1>     cmp     bh, 16 ; 8x16 font
00003209 759C                   <1>     jne     short transfer_user_fonts_4  ; invalid !
                                <1>     ;shl    dx, 4 ; * 16
                                <1>     ;shl    cx, 4 ; * 16
                                <1>     ; 02/08/2022
0000320B C1E204                 <1>     shl     edx, 4 ; byte offset
0000320E C1E104                 <1>     shl     ecx, 4 ; byte count
00003211 BF00400900             <1>     mov     edi, VGAFONT16USER
00003216 F605[4E9D0100]10       <1>     test    byte [ufont], 16  ; already loaded ?
0000321D 750A                   <1>     jnz     short transfer_user_fonts_8 ; yes
0000321F BE[10630100]           <1>     mov     esi, vgafont16
00003224 E814000000             <1>     call    transfer_user_fonts_10
                                <1> transfer_user_fonts_8:
00003229 01D7                   <1>     add     edi, edx ; char offset
                                <1>     ; 09/07/2016
                                <1>     ;and    ecx, 0FFFFh
                                <1>     ; ECX = byte count
                                <1>     ;push   ecx
0000322B 89EE                   <1>     mov     esi, ebp ; user's font buffer
                                <1>     ; 09/01/2021
0000322D 89FD                   <1>     mov     ebp, edi ; system addr for user's font
                                <1>     ; 05/01/2021
                                <1>     ;mov    edi, Cluster_Buffer  ; system buffer
0000322F E800D90000             <1>     call    transfer_from_user_buffer
                                <1>     ;pop    ecx
                                <1>     ; ecx = transfer (byte) count = character count
00003234 7206                   <1>     jc      short transfer_user_fonts_9
                                <1>     ; 05/01/2021
```

```
7118                                    <1>     ;mov    ebp, Cluster_Buffer
7119                                    <1>
7120 00003236 083D[4E9D0100]            <1>     or      byte [ufont], bh
7121                                    <1>                     ; 8x8 or 8x16 user font ready
7122                                    <1> transfer_user_fonts_9:
7123 0000323C C3                        <1>     retn
7124                                    <1>
7125                                    <1> transfer_user_fonts_10:
7126                                    <1>     ; 02/08/2022
7127                                    <1>     ; 09/01/2021
7128 0000323D 56                        <1>     push    esi
7129 0000323E 57                        <1>     push    edi
7130 0000323F 51                        <1>     push    ecx
7131                                    <1>     ;mov    cx, 64
7132                                    <1>     ; 02/08/2022
7133 00003240 31C9                      <1>     xor     ecx, ecx
7134 00003242 B140                      <1>     mov     cl, 64
7135 00003244 F3A5                      <1>     rep     movsd
7136 00003246 59                        <1>     pop     ecx
7137 00003247 5F                        <1>     pop     edi
7138 00003248 5E                        <1>     pop     esi
7139 00003249 C3                        <1>     retn
7140                                    <1>
7141                                    <1> load_text_user_pat:
7142                                    <1>     ; 02/08/2022 (TRDOS 3386 v2.0.5)
7143                                    <1>     ; 26/07/2016
7144                                    <1>     ; 09/07/2016
7145                                    <1>     ; load user defined (EGA/VGA) text fonts
7146                                    <1>     ;
7147                                    <1>     ; derived from 'Plex86/Bochs VGABios' source code
7148                                    <1>     ; vgabios-0.7a (2011)
7149                                    <1>     ; by the LGPL VGABios developers Team (2001-2008)
7150                                    <1>     ; 'vgabios.c', 'biosfn_load_text_user_pat'
7151                                    <1>
7152                                    <1>     ; biosfn_load_text_user_pat (AL,ES,BP,CX,DX,BL,BH)
7153                                    <1>
7154                                    <1>     ; get_font_access();
7155                                    <1>     ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
7156                                    <1>     ; for(i=0;i<CX;i++)
7157                                    <1>     ; {
7158                                    <1>     ;  src = BP + i * BH;
7159                                    <1>     ;  dest = blockaddr + (DX + i) * 32;
7160                                    <1>     ;  memcpyb(0xA000, dest, ES, src, BH);
7161                                    <1>     ; }
7162                                    <1>     ; release_font_access();
7163                                    <1>     ; if(AL>=0x10)
7164                                    <1>     ; {
7165                                    <1>     ; set_scan_lines(BH);
7166                                    <1>     ; }
7167                                    <1>
7168 0000324A 50                        <1>     push    eax
7169 0000324B E839000000                <1>     call    get_font_access
7170 00003250 28DB                      <1>     sub     bl, bl ; i = 0
7171                                    <1>     ; 02/08/2022
7172                                    <1>     ; ecx <= 256
7173 00003252 30ED                      <1>     xor     ch, ch
7174                                    <1> ltup_1:
7175 00003254 88D8                      <1>     mov     al, bl
7176 00003256 F6E7                      <1>     mul     bh
7177                                    <1>     ;movzx  esi, ax
7178                                    <1>     ; 02/08/2022
7179 00003258 89C6                      <1>     mov     esi, eax
7180 0000325A 01EE                      <1>     add     esi, ebp
7181 0000325C 88D8                      <1>     mov     al, bl
7182 0000325E 28E4                      <1>     sub     ah, ah
7183                                    <1>     ;add    ax, dx ; (DX + i)
7184                                    <1>     ;shl    ax, 5  ; * 32
7185                                    <1>     ; 02/08/2022
7186 00003260 01D0                      <1>     add     eax, edx
7187 00003262 C1E005                    <1>     shl     eax, 5
7188                                    <1>     ;movzx  edi, ax
7189                                    <1>     ; 02/08/2022
7190 00003265 89C7                      <1>     mov     edi, eax
7191 00003267 81C700000A00              <1>     add     edi, 0A0000h
7192 0000326D 51                        <1>     push    ecx
7193                                    <1>     ;movzx  ecx, bh
7194                                    <1>     ; 02/08/2022
7195 0000326E 88F9                      <1>     mov     cl, bh
7196 00003270 F3A4                      <1>     rep     movsb
7197 00003272 59                        <1>     pop     ecx
7198 00003273 FEC3                      <1>     inc     bl
7199 00003275 38CB                      <1>     cmp     bl, cl
7200 00003277 75DB                      <1>     jne     short ltup_1
7201                                    <1>     ;
7202 00003279 E83E000000                <1>     call    release_font_access
7203                                    <1>     ;
7204 0000327E 58                        <1>     pop     eax
7205                                    <1>     ; if(AL>=0x10)
7206 0000327F 3C10                      <1>     cmp     al, 10h
7207 00003281 7205                      <1>     jb      short ltup_2
7208                                    <1>     ; set_scan_lines(BH);
7209 00003283 E86F000000                <1>     call    set_scan_lines
7210                                    <1> ltup_2:
7211 00003288 C3                        <1>     retn
7212                                    <1>
7213                                    <1> get_font_access:
7214                                    <1>     ; 02/08/2022
7215                                    <1>     ; 09/07/2016
7216                                    <1>     ;
7217                                    <1>     ; derived from 'Plex86/Bochs VGABios' source code
7218                                    <1>     ; vgabios-0.7a (2011)
7219                                    <1>     ; by the LGPL VGABios developers Team (2001-2008)
7220                                    <1>     ; 'vgabios.c', 'get_font_access'
7221                                    <1>
7222                                    <1>     ; get_font_access()
7223 00003289 52                        <1>     push    edx
7224 0000328A 66BAC403                  <1>     mov     dx, 3C4h ; VGAREG_SEQU_ADDRESS
7225                                    <1>     ; 02/08/2022
7226 0000328E 31C0                      <1>     xor     eax, eax
7227                                    <1>     ;mov    ax, 0100h
7228 00003290 B401                      <1>     mov     ah, 1
7229                                    <1>     ; ax = 0100h
7230 00003292 66EF                      <1>     out     dx, ax
7231 00003294 66B80204                  <1>     mov     ax, 0402h
7232 00003298 66EF                      <1>     out     dx, ax
7233 0000329A 66B80407                  <1>     mov     ax, 0704h
7234 0000329E 66EF                      <1>     out     dx, ax
7235 000032A0 66B80003                  <1>     mov     ax, 0300h
7236 000032A4 66EF                      <1>     out     dx, ax
7237                                    <1>     ;mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
7238                                    <1>     ; 02/08/2022
7239 000032A6 B2CE                      <1>     mov     dl, 0CEh
7240 000032A8 66B80402                  <1>     mov     ax, 0204h
7241 000032AC 66EF                      <1>     out     dx, ax
```

```
7242 000032AE 66B80500            <1>     mov     ax, 0005h
7243 000032B2 66EF                <1>     out     dx, ax
7244 000032B4 66B80604            <1>     mov     ax, 0406h
7245 000032B8 66EF                <1>     out     dx, ax
7246 000032BA 5A                  <1>     pop     edx
7247 000032BB C3                  <1>     retn
7248                              <1>
7249                              <1> release_font_access:
7250                              <1>     ; 02/08/2022
7251                              <1>     ; 29/07/2016
7252                              <1>     ; 09/07/2016
7253                              <1>     ;
7254                              <1>     ; derived from 'Plex86/Bochs VGABios' source code
7255                              <1>     ; vgabios-0.7a (2011)
7256                              <1>     ; by the LGPL VGABios developers Team (2001-2008)
7257                              <1>     ; 'vgabios.c', 'release_font_access'
7258                              <1>
7259 000032BC 66BAC403            <1>     mov     dx, 3C4h ; VGAREG_SEQU_ADDRESS
7260                              <1>     ;mov    ax, 0100h
7261                              <1>     ; 02/08/2022
7262 000032C0 29C0                <1>     sub     eax, eax
7263 000032C2 B401                <1>     mov     ah, 1
7264                              <1>     ; ax = 0100h
7265 000032C4 66EF                <1>     out     dx, ax
7266 000032C6 66B80203            <1>     mov     ax, 0302h
7267 000032CA 66EF                <1>     out     dx, ax
7268 000032CC 66B80403            <1>     mov     ax, 0304h
7269 000032D0 66EF                <1>     out     dx, ax
7270 000032D2 66B80003            <1>     mov     ax, 0300h
7271 000032D6 66EF                <1>     out     dx, ax
7272                              <1>     ;mov    dx, 3CCh ; VGAREG_READ_MISC_OUTPUT
7273                              <1>     ; 02/08/2022
7274 000032D8 B2CC                <1>     mov     dl, 0CCh
7275 000032DA EC                  <1>     in      al, dx
7276 000032DB 2401                <1>     and     al, 01h
7277 000032DD C0E002              <1>     shl     al, 2
7278 000032E0 0C0A                <1>     or      al, 0Ah
7279 000032E2 88C4                <1>     mov     ah, al
7280 000032E4 B006                <1>     mov     al, 06h
7281                              <1>     ;mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
7282                              <1>     ; 02/08/2022
7283 000032E6 B2CE                <1>     mov     dl, 0CEh
7284 000032E8 66EF                <1>     out     dx, ax
7285 000032EA 66B80400            <1>     mov     ax, 0004h
7286 000032EE 66EF                <1>     out     dx, ax
7287 000032F0 66B80510            <1>     mov     ax, 1005h
7288 000032F4 66EF                <1>     out     dx, ax
7289 000032F6 C3                  <1>     retn
7290                              <1>
7291                              <1> set_scan_lines:
7292                              <1>     ; 02/08/2022
7293                              <1>     ; 09/07/2016
7294                              <1>     ;
7295                              <1>     ; derived from 'Plex86/Bochs VGABios' source code
7296                              <1>     ; vgabios-0.7a (2011)
7297                              <1>     ; by the LGPL VGABios developers Team (2001-2008)
7298                              <1>     ; 'vgabios.c', 'set_scan_lines'
7299                              <1>
7300                              <1>     ; set_scan_lines(lines)
7301                              <1>     ; BH = lines
7302                              <1>
7303                              <1>     ; outb(crtc_addr, 0x09);
7304 000032F7 66BAD403            <1>     mov     dx, 3D4h ; CRTC_ADDRESS = 3D4h (always)
7305 000032FB B009                <1>     mov     al, 09h
7306 000032FD EE                  <1>     out     dx, al
7307                              <1>     ; crtc_r9 = inb(crtc_addr+1);
7308                              <1>     ;inc    dx ; 3D5h
7309                              <1>     ; 02/08/2022
7310 000032FE FEC2                <1>     inc     dl
7311 00003300 EC                  <1>     in      al, dx
7312                              <1>     ; crtc_r9 = (crtc_r9 & 0xe0) | (lines - 1);
7313 00003301 24E0                <1>     and     al, 0E0h
7314 00003303 FECF                <1>     dec     bh ; lines - 1
7315 00003305 08F8                <1>     or      al, bh
7316                              <1>     ; outb(crtc_addr+1, crtc_r9);
7317 00003307 EE                  <1>     out     dx, al
7318                              <1>     ;inc    bh
7319                              <1>     ; if(lines==8)
7320                              <1>     ;cmp    bh, 8
7321 00003308 80FF07              <1>     cmp     bh, 7
7322 0000330B 7506                <1>     jne     short ssl_1
7323                              <1>     ; biosfn_set_cursor_shape(0x06,0x07);
7324 0000330D 66B90706            <1>     mov     cx, 0607h
7325 00003311 EB06                <1>     jmp     short ssl_2
7326                              <1> ssl_1:
7327                              <1>     ; biosfn_set_cursor_shape(lines-4,lines-3);
7328 00003313 88F9                <1>     mov     cl, bh ; lines - 1
7329 00003315 88CD                <1>     mov     ch, cl ; lines - 1 (16 -> 15)
7330 00003317 FECD                <1>     dec     ch  ; lines - 2 (16 -> 14)
7331                              <1> ssl_2:
7332                              <1>     ; CH = start line, CL = stop line
7333 00003319 B40A                <1>     mov     ah, 10  ; 6845 register for cursor set
7334 0000331B 66890D[37680000]    <1>     mov     [CURSOR_MODE], cx ; save in data area
7335 00003322 E877F0FFFF          <1>     call    m16     ; output cx register
7336                              <1>     ; write_word(BIOSMEM_SEG,BIOSMEM_CHAR_HEIGHT, lines);
7337 00003327 FEC7                <1>     inc     bh ; lines
7338 00003329 883D[22680000]      <1>     mov     [CHAR_HEIGHT], bh
7339                              <1>     ;  outb(crtc_addr, 0x12);
7340 0000332F 66BAD403            <1>     mov     dx, 3D4h ; CRTC_ADDRESS
7341 00003333 B012                <1>     mov     al, 12h
7342 00003335 EE                  <1>     out     dx, al
7343                              <1>     ; vde = inb(crtc_addr+1);
7344                              <1>     ;inc    dx
7345                              <1>     ; 02/08/2022
7346 00003336 FEC2                <1>     inc     dl
7347 00003338 EC                  <1>     in      al, dx
7348 00003339 88C4                <1>     mov     ah, al
7349                              <1>     ; outb(crtc_addr, 0x07);
7350                              <1>     ;dec    dx
7351                              <1>     ; 02/08/2022
7352 0000333B FECA                <1>     dec     dl
7353 0000333D B007                <1>     mov     al, 07h
7354 0000333F EE                  <1>     out     dx, al
7355                              <1>     ; ovl = inb(crtc_addr+1);
7356                              <1>     ;inc    dx
7357                              <1>     ; 02/08/2022
7358 00003340 FEC2                <1>     inc     dl
7359 00003342 EC                  <1>     in      al, dx
7360                              <1>     ; vde += (((ovl & 0x02) << 7) + ((ovl & 0x40) << 3) + 1);
7361 00003343 88E2                <1>     mov     dl, ah ; vde
7362 00003345 88C6                <1>     mov     dh, al ; ovl
7363                              <1>     ;and    ax, 02h
7364                              <1>     ;shl    ax, 7
7365                              <1>     ; 02/08/2022
```

```
7366 00003347 31C0                <1>     xor     eax, eax
7367 00003349 88F0                <1>     mov     al, dh
7368 0000334B 2402                <1>     and     al, 2
7369 0000334D C1E007              <1>     shl     eax, 7
7370                              <1>     ;mov    cx, ax ; (ovl & 0x02) << 7)
7371                              <1>     ; 02/08/2022
7372 00003350 89C1                <1>     mov     ecx, eax
7373 00003352 28E4                <1>     sub     ah, ah
7374 00003354 88F0                <1>     mov     al, dh ; ovl
7375                              <1>     ;and    ax, 40h
7376                              <1>     ;shl    ax, 3  ; (ovl & 0x40) << 3)
7377                              <1>     ;inc    ax ; + 1
7378                              <1>     ;add    ax, cx
7379                              <1>     ; 02/08/2022
7380 00003356 2440                <1>     and     al, 40h
7381 00003358 C1E003              <1>     shl     eax, 3
7382 0000335B 40                  <1>     inc     eax
7383 0000335C 01C8                <1>     add     eax, ecx
7384 0000335E 30F6                <1>     xor     dh, dh
7385                              <1>     ;add    ax, dx ; + vde
7386                              <1>     ; 02/08/2022
7387 00003360 01D0                <1>     add     eax, edx
7388                              <1>     ; rows = vde / lines;
7389 00003362 F6F7                <1>     div     bh
7390                              <1>     ;dec    al ; rows -1
7391                              <1>     ; write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, rows-1);
7392 00003364 A2[26680000]        <1>     mov     [VGA_ROWS], al ; rows (not 'rows-1' !)
7393                              <1>     ; write_word(BIOSMEM_SEG,BIOSMEM_PAGE_SIZE, rows * cols * 2);
7394                              <1>     ;mov    ah, [CRT_COLS]
7395                              <1>     ;mul    ah
7396                              <1>     ; 17/11/2020
7397 00003369 F625[20680000]      <1>     mul     byte [CRT_COLS]
7398                              <1>     ;shl    ax, 1
7399                              <1>     ; 02/08/2022
7400 0000336F D1E0                <1>     shl     eax, 1
7401 00003371 66A3[18830100]      <1>     mov     [CRT_LEN], ax
7402 00003377 C3                  <1>     retn
7403                              <1>
7404                              <1> load_text_8_14_pat:
7405                              <1>     ; 02/08/2022 (TRDOS 3386 v2.0.5)
7406                              <1>     ; 26/07/2016
7407                              <1>     ; 25/07/2016
7408                              <1>     ; 23/07/2016
7409                              <1>     ; 09/07/2016
7410                              <1>     ; load user defined (EGA/VGA) text fonts
7411                              <1>     ;
7412                              <1>     ; derived from 'Plex86/Bochs VGABios' source code
7413                              <1>     ; vgabios-0.7a (2011)
7414                              <1>     ; by the LGPL VGABios developers Team (2001-2008)
7415                              <1>     ; 'vgabios.c', 'biosfn_load_text_8_14_pat'
7416                              <1>
7417                              <1>     ; biosfn_load_text_8_14_pat (AL,BL)
7418                              <1>
7419                              <1>     ; get_font_access();
7420                              <1>     ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
7421                              <1>     ; for(i=0;i<0x100;i++)
7422                              <1>     ; {
7423                              <1>     ;  src = i * 14;
7424                              <1>     ;  dest = blockaddr + i * 32;
7425                              <1>     ;  memcpyb(0xA000, dest, 0xC000, vgafont14+src, 14);
7426                              <1>     ; }
7427                              <1>     ; release_font_access();
7428                              <1>     ; if(AL>=0x10)
7429                              <1>     ; {
7430                              <1>     ; set_scan_lines(14);
7431                              <1>     ; }
7432                              <1>
7433 00003378 50                  <1>     push    eax
7434 00003379 E80BFFFFFF          <1>     call    get_font_access
7435                              <1>
7436                              <1>     ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
7437                              <1>     ;mov    dl, bl
7438                              <1>     ;and    dl, 3
7439                              <1>     ;shl    dx, 14
7440                              <1>     ;xchg   dx, bx
7441                              <1>     ;and    dl, 4
7442                              <1>     ;shl    dx, 11
7443                              <1>     ;add    dx, bx
7444                              <1>
7445                              <1>     ;xor    dx, dx  ; blockaddr = 0
7446                              <1>     ; Always block 0 for TRDOS 386 ! (blockaddr=0)
7447                              <1>
7448 0000337E 28DB                <1>     sub     bl, bl ; i = 0
7449 00003380 B70E                <1>     mov     bh, 14
7450 00003382 BE[10550100]        <1>     mov     esi, vgafont14
7451 00003387 BF00000A00          <1>     mov     edi, 0A0000h
7452                              <1>     ; 02/08/2022
7453 0000338C 29C9                <1>     sub     ecx, ecx
7454                              <1> lt8_14_1:
7455                              <1>     ;mov    al, bl
7456                              <1>     ;mul    bh
7457                              <1>     ;movzx  esi, ax
7458                              <1>     ;add    esi, vgafont14
7459                              <1>     ;mov    al, bl
7460                              <1>     ;sub    ah, ah
7461                              <1>     ;shl    ax, 5 ; * 32
7462                              <1>     ;;add   ax, dx ; blockaddr + i * 32;
7463                              <1>     ;movzx  edi, ax ; dest
7464                              <1>     ;add    edi, 0A0000h
7465                              <1>     ;02/08/2022
7466                              <1>     ;movzx  ecx, bh
7467 0000338E 88F9                <1>     mov     cl, bh
7468 00003390 F3A4                <1>     rep     movsb
7469 00003392 83C712              <1>     add     edi, 18 ; 32 - 14
7470 00003395 FEC3                <1>     inc     bl
7471 00003397 75F5                <1>     jnz     short lt8_14_1
7472                              <1>     ;
7473 00003399 E81EFFFFFF          <1>     call    release_font_access
7474                              <1>     ;
7475 0000339E 58                  <1>     pop     eax
7476                              <1>     ; if(AL>=0x10)
7477 0000339F 3C10                <1>     cmp     al, 10h
7478 000033A1 7205                <1>     jb      short lt8_14_4
7479                              <1>     ; BH = 14
7480                              <1>     ; set_scan_lines(14);
7481 000033A3 E84FFFFFFF          <1>     call    set_scan_lines
7482                              <1> lt8_14_4:
7483 000033A8 C3                  <1>     retn
7484                              <1>
7485                              <1> load_text_8_8_pat:
7486                              <1>     ; 02/08/2022 (TRDOS 3386 v2.0.5)
7487                              <1>     ; 05/01/2021 (TRDOS 386 v2.0.3)
7488                              <1>     ; 26/07/2016
7489                              <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>
000033A9 50                     <1>     push    eax
000033AA E8DAFEFFFF             <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>
000033AF 28DB                   <1>     sub     bl, bl ; i = 0
000033B1 B708                   <1>     mov     bh, 8
                                <1>     ;mov    esi, vgafont8
000033B3 BF00000A00             <1>     mov     edi, 0A0000h
                                <1>
                                <1>     ; 02/08/2022
000033B8 29C9                   <1>     sub     ecx, ecx
                                <1>
                                <1>     ; 05/01/2021
000033BA F605[4E9D0100]80       <1>     test    byte [ufont], 80h
000033C1 7410                   <1>     jz      short lt8_8_0
                                <1>             ; user font permission (after set mode)
000033C3 F605[4E9D0100]08       <1>     test    byte [ufont], 8
000033CA 7407                   <1>     jz      short lt8_8_0
000033CC BE00500900             <1>     mov     esi, VGAFONT8USER
000033D1 EB05                   <1>     jmp     short lt8_8_1
                                <1> lt8_8_0:
000033D3 BE[104D0100]           <1>     mov     esi, vgafont8
                                <1> lt8_8_1:
                                <1>     ;mov    al, bl
                                <1>     ;mul    bh
                                <1>     ;movzx  esi, ax
                                <1>     ;add    esi, vgafont8
                                <1>     ;mov    al, bl
                                <1>     ;sub    ah, ah
                                <1>     ;shl    ax, 5 ; * 32
                                <1>     ;;add   ax, dx ; blockaddr + i * 32;
                                <1>     ;movzx  edi, ax ; dest
                                <1>     ;add    edi, 0A0000h
                                <1>     ; 02/08/2022
                                <1>     ;movzx  ecx, bh
000033D8 88F9                   <1>     mov     cl, bh
000033DA F3A4                   <1>     rep     movsb
000033DC 83C718                 <1>     add     edi, 24 ; 32 - 8
000033DF FEC3                   <1>     inc     bl
000033E1 75F5                   <1>     jnz     short lt8_8_1
                                <1>     ;
000033E3 E8D4FEFFFF             <1>     call    release_font_access
                                <1>     ;
000033E8 58                     <1>     pop     eax
                                <1>     ; if(AL>=0x10)
000033E9 3C10                   <1>     cmp     al, 10h
000033EB 7205                   <1>     jb      short lt8_8_2
                                <1>     ; BH = 8
                                <1>     ; set_scan_lines(8);
000033ED E805FFFFFF             <1>     call    set_scan_lines
                                <1> lt8_8_2:
000033F2 C3                     <1>     retn
                                <1>
                                <1> load_text_8_16_pat:
                                <1>     ; 02/08/2022 (TRDOS 3386 v2.0.5)
                                <1>     ; 05/01/2021 (TRDOS 386 v2.0.3)
                                <1>     ; 26/07/2016
                                <1>     ; 25/07/2016
                                <1>     ; 23/07/2016
                                <1>     ; 09/07/2016
                                <1>     ; load user defined (EGA/VGA) text fonts
                                <1>     ;
                                <1>     ; derived from 'Plex86/Bochs VGABios' source code
                                <1>     ; vgabios-0.7a (2011)
                                <1>     ; by the LGPL VGABios developers Team (2001-2008)
                                <1>     ; 'vgabios.c', 'biosfn_load_text_8_16_pat'
                                <1>
                                <1>     ; biosfn_load_text_8_16_pat (AL,BL)
                                <1>
                                <1>     ; get_font_access();
                                <1>     ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
                                <1>     ; for(i=0;i<0x100;i++)
                                <1>     ; {
                                <1>     ;  src = i * 16;
                                <1>     ;  dest = blockaddr + i * 32;
                                <1>     ;  memcpyb(0xA000, dest, 0xC000, vgafont16+src, 16);
                                <1>     ; }
                                <1>     ; release_font_access();
                                <1>     ; if(AL>=0x10)
                                <1>     ; {
                                <1>     ; set_scan_lines(16);
                                <1>     ; }
                                <1>
000033F3 50                     <1>     push    eax
000033F4 E890FEFFFF             <1>     call    get_font_access
                                <1>
                                <1>     ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
                                <1>     ;mov    dl, bl
```

```
7614                                <1>     ;and    dl, 3
7615                                <1>     ;shl    dx, 14
7616                                <1>     ;xchg   dx, bx
7617                                <1>     ;and    dl, 4
7618                                <1>     ;shl    dx, 11
7619                                <1>     ;add    dx, bx
7620                                <1>
7621                                <1>     ;xor    dx, dx  ; blockaddr = 0
7622                                <1>     ; Always block 0 for TRDOS 386 ! (blockaddr=0)
7623                                <1>
7624 000033F9 28DB                 <1>     sub     bl, bl ; i = 0
7625 000033FB B710                 <1>     mov     bh, 16
7626                                <1>     ;mov    esi, vgafont16
7627 000033FD BF00000A00           <1>     mov     edi, 0A0000h
7628                                <1>     ;movzx  eax, bh
7629                                <1>     ; 02/08/2022
7630 00003402 29C0                 <1>     sub     eax, eax
7631 00003404 88F8                 <1>     mov     al, bh
7632                                <1>
7633                                <1>     ; 05/01/2021
7634 00003406 F605[4E9D0100]80     <1>     test    byte [ufont], 80h
7635 0000340D 7410                 <1>     jz      short lt8_16_0
7636                                <1>             ; user font permission (after set mode)
7637 0000340F F605[4E9D0100]10     <1>     test    byte [ufont], 16
7638 00003416 7407                 <1>     jz      short lt8_16_0
7639 00003418 BE00400900           <1>     mov     esi, VGAFONT16USER
7640 0000341D EB05                 <1>     jmp     short lt8_16_1
7641                                <1> lt8_16_0:
7642 0000341F BE[10630100]         <1>     mov     esi, vgafont16
7643                                <1> lt8_16_1:
7644                                <1>     ;mov    al, bl
7645                                <1>     ;mul    bh
7646                                <1>     ;movzx  esi, ax
7647                                <1>     ;add    esi, vgafont16
7648                                <1>     ;mov    al, bl ; i
7649                                <1>     ;sub    ah, ah
7650                                <1>     ;shl    ax, 5 ; * 32
7651                                <1>     ;;add   ax, dx ; blockaddr + i * 32;
7652                                <1>     ;movzx  edi, ax ; dest
7653                                <1>     ;add    edi, 0A0000h
7654                                <1>     ;movzx  ecx, bh
7655 00003424 89C1                 <1>     mov     ecx, eax ; 16
7656 00003426 F3A4                 <1>     rep     movsb
7657 00003428 01C7                 <1>     add     edi, eax ; add edi, 16
7658 0000342A FEC3                 <1>     inc     bl
7659 0000342C 75F6                 <1>     jnz     short lt8_16_1
7660                                <1>     ;
7661 0000342E E889FEFFFF           <1>     call    release_font_access
7662                                <1>     ;
7663 00003433 58                   <1>     pop     eax
7664                                <1>     ; if(AL>=0x10)
7665 00003434 3C10                 <1>     cmp     al, 10h
7666 00003436 7205                 <1>     jb      short lt8_16_2
7667                                <1>     ; BH = 16
7668                                <1>     ; set_scan_lines(16);
7669 00003438 E8BAFEFFFF           <1>     call    set_scan_lines
7670                                <1> lt8_16_2:
7671 0000343D C3                   <1>     retn
7672                                <1>
7673                                <1> load_gfx_user_chars:
7674                                <1>     ; 08/01/2021
7675                                <1>     ; 05/01/2021 (TRDOS 386 v2.0.3)
7676                                <1>     ; 08/08/2016
7677                                <1>     ; 10/07/2016
7678                                <1>     ; Setup User-Defined Font for Graphics Mode (VGA)
7679                                <1>     ;
7680                                <1>     ; derived from 'Plex86/Bochs VGABios' source code
7681                                <1>     ; vgabios-0.7a (2011)
7682                                <1>     ; by the LGPL VGABios developers Team (2001-2008)
7683                                <1>     ; 'vgabios.c', 'biosfn_load_gfx_user_chars'
7684                                <1>
7685                                <1>     ; biosfn_load_gfx_user_chars (ES,BP,CX,BL,DL)
7686                                <1>     ; /* set 0x43 INT pointer */
7687                                <1>     ; write_word(0x0, 0x43*4, BP);
7688                                <1>     ; write_word(0x0, 0x43*4+2, ES);
7689                                <1>
7690                                <1>     ; 08/01/2021
7691                                <1>
7692                                <1>     ; BL   screen rows code: 00H = user-specified (in DL)
7693                                <1>         ;                        01H = 14 rows
7694                                <1>         ;                        02H = 25 rows
7695                                <1>         ;                        03H = 43 rows
7696                                <1>         ; CX   bytes per character definition
7697                                <1>         ; DL   (when BL=0) custom number of character rows on screen
7698                                <1>         ; EBP  address of font-definition information (user's mem space)
7699                                <1>
7700                                <1>     ; 05/01/2021
7701                                <1>     ;xor    eax, eax
7702                                <1>     ;dec    eax ; 0FFFFFFFFh (user defined fonts)
7703                                <1>     ;mov    [VGA_INT43H], eax
7704                                <1>
7705                                <1>     ; 08/01/2021
7706                                <1>     ; ebp = video font data (buffer) address
7707 0000343E 892D[2A830100]       <1>     mov     [VGA_INT43H], ebp
7708                                <1>
7709                                <1>     ; switch (BL) {
7710                                <1>     ; case 0:
7711                                <1>     ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, DL-1);
7712                                <1>     ;    break;
7713 00003444 20DB                 <1>     and     bl, bl
7714 00003446 7508                 <1>     jnz     short l_gfx_uc_1
7715 00003448 8815[26680000]       <1>     mov     [VGA_ROWS], dl  ; not DL-1 !
7716 0000344E EB23                 <1>     jmp     short l_gfx_uc_4
7717                                <1> l_gfx_uc_1:
7718                                <1>     ; case 1:
7719                                <1>     ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 13);
7720                                <1>     ;    break;
7721 00003450 FECB                 <1>     dec     bl
7722 00003452 7509                 <1>     jnz     short l_gfx_uc_2
7723                                <1>     ; bl = 1
7724 00003454 C605[26680000]0E     <1>     mov     byte [VGA_ROWS], 14  ; not 13 !
7725 0000345B EB16                 <1>     jmp     short l_gfx_uc_4
7726                                <1> l_gfx_uc_2:
7727 0000345D FECB                 <1>     dec     bl
7728 0000345F 740B                 <1>     jz      short l_gfx_uc_3 ; bl = 2
7729 00003461 FECB                 <1>     dec     bl
7730 00003463 750E                 <1>     jnz     short l_gfx_uc_4 ; bl > 3
7731                                <1>     ; bl = 3
7732                                <1>     ; case 3:
7733                                <1>     ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 42);
7734                                <1>     ;    break;
7735 00003465 C605[26680000]2B     <1>     mov     byte [VGA_ROWS], 43  ; not 42 !
7736                                <1> l_gfx_uc_3:
7737                                <1>     ; case 2:
```

```
                            <1>     ; default:
                            <1>     ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
                            <1>     ;    break;
                            <1>     ; bl = 2 or bl > 3
0000346C C605[26680000]19   <1>     mov    byte [VGA_ROWS], 25  ; not 24 !
                            <1>     ; }
                            <1> l_gfx_uc_4:
                            <1>     ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, CX);
00003473 880D[22680000]     <1>     mov    [CHAR_HEIGHT], cl
                            <1>     ; }
00003479 C3                 <1>     retn
                            <1>
                            <1> load_gfx_8_14_chars:
                            <1>     ; 08/08/2016
                            <1>     ; 10/07/2016
                            <1>     ; Setup ROM 8x14 Font for Graphics Mode (VGA)
                            <1>     ;
                            <1>     ; derived from 'Plex86/Bochs VGABios' source code
                            <1>     ; vgabios-0.7a (2011)
                            <1>     ; by the LGPL VGABios developers Team (2001-2008)
                            <1>     ; 'vgabios.c', 'biosfn_load_gfx_8_14_chars'
                            <1>
                            <1>     ; biosfn_load_gfx_8_14_chars (BL)
                            <1>     ; /* set 0x43 INT pointer */
                            <1>     ; write_word(0x0, 0x43*4, &vgafont14);
                            <1>     ; write_word(0x0, 0x43*4+2, 0xC000);
0000347A C705[2A830100]-    <1>     mov    dword [VGA_INT43H], vgafont14
00003480 [10550100]         <1>
                            <1>
                            <1>     ; BL    screen rows code: 00H = user-specified (in DL)
                            <1>     ;                         01H = 14 rows
                            <1>     ;                         02H = 25 rows
                            <1>     ;                         03H = 43 rows
                            <1>     ; DL    (when BL=0) custom number of char rows on screen
                            <1>
                            <1>     ; switch (BL) {
                            <1>     ; case 0:
                            <1>     ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, DL-1);
                            <1>     ;    break;
00003484 20DB               <1>     and    bl, bl
00003486 7508               <1>     jnz    short l_gfx_8_14c_1
00003488 8815[26680000]     <1>     mov    [VGA_ROWS], dl  ; not DL-1 !
0000348E EB23               <1>     jmp    short l_gfx_8_14c_4
                            <1> l_gfx_8_14c_1:
                            <1>     ; case 1:
                            <1>     ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 13);
                            <1>     ;    break;
00003490 FECB               <1>     dec    bl
00003492 7509               <1>     jnz    short l_gfx_8_14c_2
                            <1>     ; bl = 1
00003494 C605[26680000]0E   <1>     mov    byte [VGA_ROWS], 14  ; not 13 !
0000349B EB16               <1>     jmp    short l_gfx_8_14c_4
                            <1> l_gfx_8_14c_2:
0000349D FECB               <1>     dec    bl
0000349F 740B               <1>     jz     short l_gfx_8_14c_3 ; bl = 2
000034A1 FECB               <1>     dec    bl
000034A3 750E               <1>     jnz    short l_gfx_8_14c_4 ; bl > 3
                            <1>     ; bl = 3
                            <1>     ; case 3:
                            <1>     ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 42);
                            <1>     ;    break;
000034A5 C605[26680000]2B   <1>     mov    byte [VGA_ROWS], 43  ; not 42 !
                            <1> l_gfx_8_14c_3:
                            <1>     ; case 2:
                            <1>     ; default:
                            <1>     ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
                            <1>     ;    break;
                            <1>     ; bl = 2 or bl > 3
000034AC C605[26680000]19   <1>     mov    byte [VGA_ROWS], 25  ; not 24 !
                            <1>     ; }
                            <1> l_gfx_8_14c_4:
                            <1>     ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, 14);
000034B3 C605[22680000]0E   <1>       mov     byte [CHAR_HEIGHT], 14
                            <1>     ; }
000034BA C3                 <1>     retn
                            <1>
                            <1> load_gfx_8_8_chars:
                            <1>     ; 08/08/2016
                            <1>     ; 10/07/2016
                            <1>     ; Setup ROM 8x14 Font for Graphics Mode (VGA)
                            <1>     ;
                            <1>     ; derived from 'Plex86/Bochs VGABios' source code
                            <1>     ; vgabios-0.7a (2011)
                            <1>     ; by the LGPL VGABios developers Team (2001-2008)
                            <1>     ; 'vgabios.c', 'biosfn_load_gfx_8_8_dd_chars'
                            <1>
                            <1>     ; biosfn_load_gfx_8_8_dd_chars (BL)
                            <1>     ; /* set 0x43 INT pointer */
                            <1>     ; write_word(0x0, 0x43*4, &vgafont8);
                            <1>     ; write_word(0x0, 0x43*4+2, 0xC000);
000034BB C705[2A830100]-    <1>     mov    dword [VGA_INT43H], vgafont8
000034C1 [104D0100]         <1>
                            <1>
                            <1>     ; BL    screen rows code: 00H = user-specified (in DL)
                            <1>     ;                         01H = 14 rows
                            <1>     ;                         02H = 25 rows
                            <1>     ;                         03H = 43 rows
                            <1>     ; DL    (when BL=0) custom number of char rows on screen
                            <1>
                            <1>     ; switch (BL) {
                            <1>     ; case 0:
                            <1>     ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, DL-1);
                            <1>     ;    break;
000034C5 20DB               <1>     and    bl, bl
000034C7 7508               <1>     jnz    short l_gfx_8_8c_1
000034C9 8815[26680000]     <1>     mov    [VGA_ROWS], dl  ; not DL-1 !
000034CF EB23               <1>     jmp    short l_gfx_8_8c_4
                            <1> l_gfx_8_8c_1:
                            <1>     ; case 1:
                            <1>     ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 13);
                            <1>     ;    break;
000034D1 FECB               <1>     dec    bl
000034D3 7509               <1>     jnz    short l_gfx_8_8c_2
                            <1>     ; bl = 1
000034D5 C605[26680000]0E   <1>     mov    byte [VGA_ROWS], 14  ; not 13 !
000034DC EB16               <1>     jmp    short l_gfx_8_8c_4
                            <1> l_gfx_8_8c_2:
000034DE FECB               <1>     dec    bl
000034E0 740B               <1>     jz     short l_gfx_8_8c_3 ; bl = 2
000034E2 FECB               <1>     dec    bl
000034E4 750E               <1>     jnz    short l_gfx_8_8c_4 ; bl > 3
                            <1>     ; bl = 3
                            <1>     ; case 3:
                            <1>     ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 42);
```

```
7860                                <1>     ;    break;
7861 000034E6 C605[26680000]2B      <1>     mov    byte [VGA_ROWS], 43  ; not 42 !
7862                                <1> l_gfx_8_8c_3:
7863                                <1>     ; case 2:
7864                                <1>     ; default:
7865                                <1>     ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
7866                                <1>     ;    break;
7867                                <1>     ; bl = 2 or bl > 3
7868 000034ED C605[26680000]19      <1>     mov    byte [VGA_ROWS], 25  ; not 24 !
7869                                <1>     ; }
7870                                <1> l_gfx_8_8c_4:
7871                                <1>     ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, 8);
7872 000034F4 C605[22680000]08      <1>       mov     byte [CHAR_HEIGHT], 8
7873                                <1>     ; }
7874 000034FB C3                    <1>     retn
7875                                <1>
7876                                <1> load_gfx_8_16_chars:
7877                                <1>     ; 08/08/2016
7878                                <1>     ; 10/07/2016
7879                                <1>     ; Setup ROM 8x14 Font for Graphics Mode (VGA)
7880                                <1>     ;
7881                                <1>     ; derived from 'Plex86/Bochs VGABios' source code
7882                                <1>     ; vgabios-0.7a (2011)
7883                                <1>     ; by the LGPL VGABios developers Team (2001-2008)
7884                                <1>     ; 'vgabios.c', 'biosfn_load_gfx_8_16_chars'
7885                                <1>
7886                                <1>     ; biosfn_load_gfx_8_16_chars (BL)
7887                                <1>     ; /* set 0x43 INT pointer */
7888                                <1>     ; write_word(0x0, 0x43*4, &vgafont16);
7889                                <1>     ; write_word(0x0, 0x43*4+2, 0xC000);
7890 000034FC C705[2A830100]-      <1>     mov    dword [VGA_INT43H], vgafont16
7890 00003502 [10630100]            <1>
7891                                <1>
7892                                <1>     ; BL    screen rows code: 00H = user-specified (in DL)
7893                                <1>        ;                         01H = 14 rows
7894                                <1>        ;                         02H = 25 rows
7895                                <1>        ;                         03H = 43 rows
7896                                <1>        ; DL    (when BL=0) custom number of char rows on screen
7897                                <1>
7898                                <1>     ; switch (BL) {
7899                                <1>     ; case 0:
7900                                <1>     ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, DL-1);
7901                                <1>     ;    break;
7902 00003506 20DB                  <1>     and    bl, bl
7903 00003508 7508                  <1>     jnz    short l_gfx_8_16c_1
7904 0000350A 8815[26680000]        <1>     mov    [VGA_ROWS], dl  ; not DL-1 !
7905 00003510 EB23                  <1>     jmp    short l_gfx_8_16c_4
7906                                <1> l_gfx_8_16c_1:
7907                                <1>     ; case 1:
7908                                <1>     ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 13);
7909                                <1>     ;    break;
7910 00003512 FECB                  <1>     dec    bl
7911 00003514 7509                  <1>     jnz    short l_gfx_8_16c_2
7912                                <1>     ; bl = 1
7913 00003516 C605[26680000]0E      <1>     mov    byte [VGA_ROWS], 14  ; not 13 !
7914 0000351D EB16                  <1>     jmp    short l_gfx_8_16c_4
7915                                <1> l_gfx_8_16c_2:
7916 0000351F FECB                  <1>     dec    bl
7917 00003521 740B                  <1>     jz     short l_gfx_8_16c_3 ; bl = 2
7918 00003523 FECB                  <1>     dec    bl
7919 00003525 750E                  <1>     jnz    short l_gfx_8_16c_4 ; bl > 3
7920                                <1>     ; bl = 3
7921                                <1>     ; case 3:
7922                                <1>     ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 42);
7923                                <1>     ;    break;
7924 00003527 C605[26680000]2B      <1>     mov    byte [VGA_ROWS], 43  ; not 42 !
7925                                <1> l_gfx_8_16c_3:
7926                                <1>     ; case 2:
7927                                <1>     ; default:
7928                                <1>     ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
7929                                <1>     ;    break;
7930                                <1>     ; bl = 2 or bl > 3
7931 0000352E C605[26680000]19      <1>     mov    byte [VGA_ROWS], 25  ; not 24 !
7932                                <1>     ; }
7933                                <1> l_gfx_8_16c_4:
7934                                <1>     ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, 16);
7935 00003535 C605[22680000]10      <1>       mov     byte [CHAR_HEIGHT], 16
7936                                <1>     ; }
7937 0000353C C3                    <1>     retn
7938                                <1>
7939                                <1> get_font_info:
7940                                <1>     ; 03/08/2022 (TRDOS 386 v2.0.5)
7941                                <1>     ; 08/01/2021 (TRDOS 386 v2.0.3)
7942                                <1>     ; 19/09/2016
7943                                <1>     ; 08/08/2016
7944                                <1>     ; 10/07/2016
7945                                <1>     ; Get Current Character Generator Info (VGA)
7946                                <1>     ;
7947                                <1>     ; derived from 'Plex86/Bochs VGABios' source code
7948                                <1>     ; vgabios-0.7a (2011)
7949                                <1>     ; by the LGPL VGABios developers Team (2001-2008)
7950                                <1>     ; 'vgabios.c', 'biosfn_get_font_info'
7951                                <1>
7952                                <1>     ; Modified for TRDOS 386 !
7953                                <1>     ;
7954                                <1>     ; INPUT ->
7955                                <1>     ;    AX = 1130h
7956                                <1>     ;    BL = 0 -> Get info for current VGA font
7957                                <1>     ;              (BH = unused)
7958                                <1>     ;    19/09/2016
7959                                <1>     ;    BL > 0 -> Get requested character font data
7960                                <1>     ;        BL = 1 -> vgafont8
7961                                <1>     ;         BL = 2 -> vgafont14
7962                                <1>     ;        BL = 3 -> vgafont16
7963                                <1>     ;       ;08/01/2021
7964                                <1>     ;        BL = 4 -> user defined 8x8 font
7965                                <1>     ;        BL = 5 -> user defined 8x14 font
7966                                <1>     ;        BL = 6 -> user defined 8x16 font
7967                                <1>     ;        BL > 6 -> Invalid function (for now!)
7968                                <1>     ;        BH = ASCII code of the first character
7969                                <1>     ;        ECX = Number of characters from the 1st char
7970                                <1>     ;        ECX >= 256 -> All (256-BH) characters
7971                                <1>     ;        ECX = 0 -> All characters (BH = unused)
7972                                <1>     ;         EDX = User's Buffer Address
7973                                <1>     ; OUTPUT ->
7974                                <1>     ;    AL = height (scanlines), bytes per character
7975                                <1>     ;    AH = screen rows
7976                                <1>     ;    Byte 16-23 of EAX = number of columns
7977                                <1>     ;    Byte 24-31 of EAX =
7978                                <1>     ;         0 -> default font (not configured yet)
7979                                <1>     ;         0FFh -> user defined font
7980                                <1>     ;         14 = vgafont14
7981                                <1>     ;          8 = vgafont8
7982                                <1>     ;         16 = vgafont16
```

```
                                <1>     ;   If BL input > 0 ->
                                <1>     ;       EAX = Actual transfer count
                                <1>     ;
0000353D 20DB                   <1>     and     bl, bl
0000353F 740F                   <1>     jz      short gfi_1
                                <1>     ; invalid function (input)
                                <1>     ; 08/01/2021
00003541 80FB04                 <1>     cmp     bl, 4
00003544 7263                   <1>     jb      short gfi_5
00003546 7441                   <1>     je      short gfi_3
00003548 80FB06                 <1>     cmp     bl, 6
0000354B 744C                   <1>     je      short gfi_4
                                <1>     ; bh = 5 or bh > 6
                                <1> gfi_0:
0000354D 31C0                   <1>     xor     eax, eax ; 0
0000354F C3                     <1>     retn
                                <1> gfi_1:
00003550 A0[22680000]           <1>     mov     al, [CHAR_HEIGHT]
00003555 8A25[26680000]         <1>     mov     ah, [VGA_ROWS]
0000355B C1E010                 <1>     shl     eax, 16
0000355E A0[20680000]           <1>     mov     al, [CRT_COLS]
00003563 8B0D[2A830100]         <1>     mov     ecx, [VGA_INT43H]
00003569 21C9                   <1>     and     ecx, ecx
0000356B 7418                   <1>     jz      short gfi_2 ; 0 = default font
                                <1>     ; 08/01/2021
0000356D FECC                   <1>     dec     ah ; 0FFh
0000356F 81F900400900           <1>     cmp     ecx, VGAFONT16USER
00003575 740E                   <1>     je      short gfi_2
00003577 81F900500900           <1>     cmp     ecx, VGAFONT8USER
0000357D 7406                   <1>     je      short gfi_2
0000357F 8A25[22680000]         <1>     mov     ah, [CHAR_HEIGHT] ; font size = height
                                <1> gfi_2:
00003585 C1C010                 <1>     rol     eax, 16
00003588 C3                     <1>     retn
                                <1> gfi_3:
                                <1>     ; 08/01/2021
00003589 F605[4E9D0100]08       <1>     test    byte [ufont], 08h ; 8x8 user font
00003590 74BB                   <1>     jz      short gfi_0 ; not loaded !
00003592 BE00500900             <1>     mov     esi, VGAFONT8USER ; *
                                <1>     ;mov    bl, 8
                                <1>     ;jmp    short gfi_8
00003597 EB4A                   <1>     jmp     short gfi_10
                                <1> gfi_4:
                                <1>     ; 08/01/2021
00003599 F605[4E9D0100]10       <1>     test    byte [ufont], 10h ; 8x16 user font
000035A0 74AB                   <1>     jz      short gfi_0 ; not loaded !
000035A2 BE00400900             <1>     mov     esi, VGAFONT16USER ; *
000035A7 EB15                   <1>     jmp     short gfi_7
                                <1> gfi_5:
000035A9 80FB02                 <1>     cmp     bl, 2
000035AC 7230                   <1>     jb      short gfi_9
000035AE 7709                   <1>     ja      short gfi_6
                                <1>     ; BL = 2 -> vgafont14
000035B0 BE[10550100]           <1>     mov     esi, vgafont14 ; *
000035B5 B30E                   <1>     mov     bl, 14
000035B7 EB07                   <1>     jmp     short gfi_8
                                <1> gfi_6:
                                <1>     ; BL = 3 -> vgafont16
000035B9 BE[10630100]           <1>     mov     esi, vgafont16 ; *
                                <1> gfi_7:
000035BE B310                   <1>     mov     bl, 16
                                <1> gfi_8:
000035C0 89D7                   <1>     mov     edi, edx ; **
000035C2 09C9                   <1>     or      ecx, ecx
000035C4 7421                   <1>     jz      short gfi_11 ; all chars from the 00h
                                <1>     ;mov    al, bh ; character index
                                <1>     ; 03/08/2022
000035C6 0FB6C7                 <1>     movzx   eax, bh
000035C9 F6E3                   <1>     mul     bl ; char index * char height/size
                                <1>     ;movzx  edx, ax
                                <1>     ; 03/08/2022
                                <1>     ;add    esi, edx ; *
000035CB 01C6                   <1>     add     esi, eax
000035CD 31D2                   <1>     xor     edx, edx
000035CF FECA                   <1>     dec     dl
                                <1>     ; edx = 0FFh = 255
                                <1>     ;mov    dx, 255
000035D1 28FA                   <1>     sub     dl, bh
                                <1>     ;inc    dx
                                <1>     ; 03/08/2022
000035D3 42                     <1>     inc     edx
000035D4 39D1                   <1>     cmp     ecx, edx
000035D6 770F                   <1>     ja      short gfi_11
000035D8 7411                   <1>     je      short gfi_12
000035DA 89D1                   <1>     mov     ecx, edx
000035DC EB0D                   <1>     jmp     short gfi_12
                                <1> gfi_9:
                                <1>     ;BL = 1 -> vgafont8
000035DE BE[104D0100]           <1>     mov     esi, vgafont8 ; *
                                <1> gfi_10:
000035E3 B308                   <1>     mov     bl, 8
000035E5 EBD9                   <1>     jmp     short gfi_8
                                <1> gfi_11:
                                <1>     ;mov    ecx, 256
                                <1>     ; 03/08/2022
000035E7 29C9                   <1>     sub     ecx, ecx
000035E9 FEC5                   <1>     inc     ch
                                <1>     ; ecx = 100h
                                <1> gfi_12:
                                <1>     ; 08/01/2021
000035EB 89C8                   <1>     mov     eax, ecx ; character count
000035ED 30FF                   <1>     xor     bh, bh
000035EF 66F7E3                 <1>     mul     bx ; char count * char height/size
000035F2 89C1                   <1>     mov     ecx, eax
                                <1>
                                <1>     ; ESI = source address in system space
                                <1>     ; EDI = user's buffer address
                                <1>     ; ECX = transfer (byte) count
000035F4 E8F1D40000             <1>     call    transfer_to_user_buffer
000035F9 89C8                   <1>     mov     eax, ecx ; actual transfer count
000035FB C3                     <1>     retn
                                <1>
                                <1> set_all_palette_reg:
                                <1>     ; 03/08/2022
                                <1>     ; 10/08/2016
                                <1>     ; Set All Palette Registers and Overscan
                                <1>     ; EDX = Address of 17 bytes;
                                <1>     ; an rgbRGB value for each of 16 palette
                                <1>     ; registers plus one for the border.
                                <1>
000035FC 89D6                   <1>     mov     esi, edx ; user buffer
                                <1>     ;mov    ecx, 17
                                <1>     ; 03/08/2022
000035FE 29C9                   <1>     sub     ecx, ecx
```

```
8107 00003600 B111              <1>     mov     cl, 17
8108 00003602 89E7              <1>     mov     edi, esp
8109 00003604 83EC14            <1>     sub     esp, 20
8110 00003607 E828D50000        <1>     call    transfer_from_user_buffer
8111                            <1>     ;jc     VIDEO_RETURN
8112                            <1>
8113 0000360C 66BADA03          <1>     mov     dx, 3DAh ; VGAREG_ACTL_RESET
8114 00003610 EC                <1>     in      al, dx
8115                            <1>     ;mov    cl, 0
8116                            <1>     ; 03/08/2022
8117 00003611 28C9              <1>     sub     cl, cl
8118                            <1>     ;mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
8119                            <1>     ; 03/08/2022
8120 00003613 B2C0              <1>     mov     dl, 0C0h
8121                            <1> set_palette_loop:
8122 00003615 88C8              <1>     mov     al, cl
8123 00003617 EE                <1>     out     dx, al
8124 00003618 8A07              <1>     mov     al, [edi]
8125 0000361A EE                <1>     out     dx, al
8126 0000361B 47                <1>     inc     edi
8127 0000361C FEC1              <1>     inc     cl
8128 0000361E 80F910            <1>     cmp     cl, 10h
8129 00003621 75F2              <1>     jne     short set_palette_loop
8130 00003623 B011              <1>     mov     al, 11h
8131 00003625 EE                <1>     out     dx, al
8132 00003626 8A07              <1>     mov     al, [edi]
8133 00003628 EE                <1>     out     dx, al
8134 00003629 B020              <1>     mov     al, 20h
8135 0000362B EE                <1>     out     dx, al
8136                            <1>     ; ifdef VBOX
8137                            <1>     ;mov    dx, 3DAh ; VGAREG_ACTL_RESET
8138                            <1>     ; 03/08/2022
8139 0000362C B2DA              <1>     mov     dl, 0DAh
8140 0000362E EC                <1>     in      al, dx
8141                            <1>     ; endif ; VBOX
8142 0000362F 83C414            <1>     add     esp, 20
8143 00003632 E941E5FFFF        <1>     jmp     VIDEO_RETURN
8144                            <1>
8145                            <1>     ; 07/08/2022
8146                            <1> toggle_intensity:
8147                            <1>     ; 03/08/2022
8148                            <1>     ; 10/08/2016
8149                            <1>     ; Select Foreground Blink or Bold Background
8150                            <1>     ; BL = 00h = enable bold backgrounds
8151                            <1>     ;           (16 background colors)
8152                            <1>        ;      01h = enable blinking foreground
8153                            <1>     ;           (8 background colors)
8154                            <1>
8155 00003637 66BADA03          <1>     mov     dx, 3DAh ; VGAREG_ACTL_RESET
8156 0000363B EC                <1>     in      al, dx
8157                            <1>     ;mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
8158                            <1>     ; 03/08/2022
8159 0000363C B2C0              <1>     mov     dl, 0C0h
8160 0000363E B010              <1>     mov     al, 10h
8161 00003640 EE                <1>     out     dx, al
8162                            <1>     ;mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
8163                            <1>     ; 03/08/2022
8164 00003641 FEC2              <1>     inc     dl ; dx = 3C1h
8165 00003643 EC                <1>     in      al, dx
8166 00003644 24F7              <1>     and     al, 0F7h
8167 00003646 80E301            <1>     and     bl, 01h
8168 00003649 C0E303            <1>     shl     bl, 3
8169 0000364C 08D8              <1>     or      al, bl
8170                            <1>     ;mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
8171                            <1>     ; 03/08/2022
8172 0000364E FECA              <1>     dec     dl ; dx = 3C0h
8173 00003650 EE                <1>     out     dx, al
8174 00003651 B020              <1>     mov     al, 20h
8175 00003653 EE                <1>     out     dx, al
8176                            <1>     ; ifdef VBOX
8177                            <1>     ;mov    dx, 3DAh ; VGAREG_ACTL_RESET
8178                            <1>     ; 03/08/2022
8179 00003654 B2DA              <1>     mov     dl, 0DAh
8180 00003656 EC                <1>     in      al, dx
8181                            <1>     ; endif ; VBOX
8182 00003657 E91CE5FFFF        <1>     jmp     VIDEO_RETURN
8183                            <1>
8184                            <1>     ; 07/08/2022
8185                            <1> vga_palf_unknown:
8186 0000365C 29C0              <1>     sub     eax, eax ; 0 = invalid function
8187 0000365E E91AE5FFFF        <1>         jmp     _video_return
8188                            <1>
8189                            <1>     ; 07/08/2022
8190                            <1> vga_palf_101B:
8191 00003663 3C1B              <1>     cmp     al, 1Bh
8192                            <1>     ;jne    short vga_palf_unknown
8193 00003665 77F5              <1>     ja      short vga_palf_unknown
8194                            <1>
8195 00003667 E810F6FFFF        <1>     call    gray_scale_summing
8196 0000366C E907E5FFFF        <1>         jmp     VIDEO_RETURN
8197                            <1>
8198                            <1> vga_pal_funcs:
8199                            <1>     ; 07/08/2022
8200                            <1>     ; 10/08/2016
8201                            <1>     ; VGA Palette functions
8202                            <1>     ;
8203                            <1>     ; derived from 'Plex86/Bochs VGABios' source code
8204                            <1>     ; vgabios-0.7a (2011)
8205                            <1>     ; by the LGPL VGABios developers Team (2001-2008)
8206                            <1>     ; 'vgabios.c', 'vgarom.asm'
8207                            <1>
8208 00003671 3C00              <1>     cmp     al, 0
8209 00003673 7467              <1>         je      short set_single_palette_reg
8210                            <1> vga_palf_1002:
8211 00003675 3C02              <1>     cmp     al, 2
8212 00003677 7483              <1>         je      short set_all_palette_reg
8213                            <1>     ; 07/08/2022
8214 00003679 7702              <1>     ja      short vga_palf_1003
8215 0000367B EB5D              <1>     jmp     short set_overscan_border_color
8216                            <1>     ; 07/08/2022
8217                            <1> ;vga_palf_1001:
8218                            <1> ;   cmp     al, 1
8219                            <1> ;   je      short set_overscan_border_color
8220                            <1> vga_palf_1003:
8221 0000367D 3C03              <1>     cmp     al, 3
8222 0000367F 74B6              <1>     je      short toggle_intensity
8223                            <1> vga_palf_1007:
8224 00003681 3C07              <1>     cmp     al, 7
8225 00003683 747F              <1>         je      short get_single_palette_reg
8226 00003685 72D5              <1>     jb      short vga_palf_unknown
8227                            <1> vga_palf_1008:
8228 00003687 3C08              <1>     cmp     al, 8
8229 00003689 7477              <1>         je      short read_overscan_border_color
8230                            <1> vga_palf_1009:
```

```
8231 0000368B 3C09              <1>     cmp     al, 9
8232                            <1>     ;je     short get_all_palette_reg
8233                            <1>     ; 07/08/2022
8234 0000368D 7505              <1>     jne     short vga_palf_1010
8235 0000368F E966010000        <1>     jmp     get_all_palette_reg
8236                            <1> vga_palf_1010:
8237 00003694 3C10              <1>     cmp     al, 10h
8238                            <1>     ;je     short set_single_dac_reg
8239                            <1>     ; 07/08/2022
8240 00003696 7707              <1>     ja      short vga_palf_1012
8241 00003698 72C2              <1>     jb      short vga_palf_unknown
8242 0000369A E908010000        <1>     jmp     set_single_dac_reg
8243                            <1> vga_palf_1012:
8244 0000369F 3C12              <1>     cmp     al, 12h
8245                            <1>     ;je     short set_all_dac_reg
8246                            <1>     ; 07/08/2022
8247 000036A1 7707              <1>     ja      short vga_palf_1013
8248 000036A3 72B7              <1>     jb      short vga_palf_unknown
8249 000036A5 E916010000        <1>     jmp     set_all_dac_reg
8250                            <1> vga_palf_1013:
8251 000036AA 3C13              <1>     cmp     al, 13h
8252                            <1>     ;je     short select_video_dac_color_page
8253                            <1>     ; 07/08/2022
8254 000036AC 7505              <1>     jne     short vga_palf_1015
8255 000036AE E992010000        <1>     jmp     select_video_dac_color_page
8256                            <1> vga_palf_1015:
8257 000036B3 3C15              <1>     cmp     al, 15h
8258                            <1>     ;je     short read_single_dac_reg
8259                            <1>     ; 07/08/2022
8260 000036B5 7707              <1>     ja      short vga_palf_1017
8261 000036B7 72A3              <1>     jb      short vga_palf_unknown
8262 000036B9 E98D000000        <1>     jmp     read_single_dac_reg
8263                            <1> vga_palf_1017:
8264 000036BE 3C17              <1>     cmp     al, 17h
8265                            <1>     ;je     short read_all_dac_reg
8266                            <1>     ; 07/08/2022
8267 000036C0 7707              <1>     ja      short vga_palf_1018
8268 000036C2 7298              <1>     jb      short vga_palf_unknown
8269 000036C4 E9A0000000        <1>     jmp     read_all_dac_reg
8270                            <1> vga_palf_1018:
8271 000036C9 3C18              <1>     cmp     al, 18h
8272 000036CB 7464              <1>         je  short set_pel_mask
8273                            <1> vga_palf_1019:
8274 000036CD 3C19              <1>     cmp     al, 19h
8275 000036CF 746C              <1>         je  short read_pel_mask
8276                            <1> vga_palf_101A:
8277 000036D1 3C1A              <1>     cmp     al, 1Ah
8278                            <1>         ;je short read_video_dac_state
8279                            <1>     ; 07/08/2022
8280 000036D3 758E              <1>     jne     short vga_palf_101B
8281 000036D5 E9AD010000        <1>     jmp     read_video_dac_state
8282                            <1>
8283                            <1>     ; 07/08/2022
8284                            <1> set_overscan_border_color:
8285                            <1>     ; 10/08/2016
8286                            <1>     ; Set Overscan/Border Color Register
8287                            <1>     ; BH = 6-bit RGB color to display
8288                            <1>     ;      for that attribute
8289                            <1>
8290 000036DA B311              <1>     mov     bl, 11h
8291                            <1>     ; 07/08/2022
8292                            <1>     ;jmp    short set_single_palette_reg
8293                            <1>
8294                            <1> set_single_palette_reg:
8295                            <1>     ; 03/08/2022 (TRDOS 386 v2.0.5)
8296                            <1>     ; 12/04/2021 (TRDOS 386 v2.0.3, 32 bit push/pop)
8297                            <1>     ; 10/08/2016
8298                            <1>     ; Set One Palette Register
8299                            <1>     ; BL = register number to set
8300                            <1>     ;     (a 4-bit attribute nibble: 00h-0Fh)
8301                            <1>     ; BH = 6-bit RGB color to display
8302                            <1>     ;      for that attribute
8303                            <1>
8304 000036DC 80FB14            <1>     cmp     bl, 14h
8305                            <1>     ;;ja    short no_actl_reg1
8306                            <1>     ;ja     VIDEO_RETURN
8307                            <1>     ; 03/08/2022
8308 000036DF 7605              <1>     jna     short sspr_1
8309 000036E1 E992E4FFFF        <1>     jmp     VIDEO_RETURN
8310                            <1> sspr_1:
8311                            <1>     ;push   ax
8312                            <1>     ;push   dx
8313                            <1>     ; 12/04/2021
8314 000036E6 50                <1>     push    eax
8315 000036E7 52                <1>     push    edx
8316 000036E8 66BADA03          <1>     mov     dx, 3DAh ; VGAREG_ACTL_RESET
8317 000036EC EC                <1>     in      al, dx
8318                            <1>     ;mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
8319                            <1>     ; 03/08/2022
8320 000036ED B2C0              <1>     mov     dl, 0C0h
8321 000036EF 88D8              <1>     mov     al, bl
8322 000036F1 EE                <1>     out     dx, al
8323 000036F2 88F8              <1>     mov     al, bh
8324 000036F4 EE                <1>     out     dx, al
8325 000036F5 B020              <1>     mov     al, 20h
8326 000036F7 EE                <1>     out     dx, al
8327                            <1>     ; ifdef VBOX
8328                            <1>     ;mov    dx, 3DAh ; VGAREG_ACTL_RESET
8329                            <1>     ; 03/08/2022
8330 000036F8 B2DA              <1>     mov     dl, 0DAh
8331 000036FA EC                <1>     in      al, dx
8332                            <1>     ; endif ; VBOX
8333                            <1>     ;pop    dx
8334                            <1>     ;pop    ax
8335                            <1>     ; 12/04/2021
8336 000036FB 5A                <1>     pop     edx
8337 000036FC 58                <1>     pop     eax
8338                            <1> ;no_actl_reg1:
8339 000036FD E976E4FFFF        <1>     jmp     VIDEO_RETURN
8340                            <1>
8341                            <1>     ; 07/08/2022
8342                            <1> read_overscan_border_color:
8343                            <1>     ; 10/08/2016
8344                            <1>     ; Read Overscan Register
8345                            <1>     ; OUTPUT:
8346                            <1>     ; BH = current rgbRGB value
8347                            <1>     ;      of the overscan/border register
8348                            <1>
8349 00003702 B311              <1>     mov     bl, 11h
8350                            <1>     ; 07/08/2022
8351                            <1>     ;jmp    short get_single_palette_reg
8352                            <1>
8353                            <1> get_single_palette_reg:
8354                            <1>     ; 03/08/2022
```

```
                            <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>
00003704 80FB14             <1>     cmp     bl, 14h
                            <1>     ;;ja    short no_actl_reg2
                            <1>     ;ja     VIDEO_RETURN
                            <1>     ; 03/08/2022
00003707 7605               <1>     jna     short gspr_1
00003709 E96AE4FFFF         <1>     jmp     VIDEO_RETURN
                            <1> gspr_1:
0000370E 66BADA03           <1>     mov     dx, 3DAh ; VGAREG_ACTL_RESET
                            <1>     ; 03/08/2022
00003712 B2DA               <1>     mov     dl, 0DAh
00003714 EC                 <1>     in      al, dx
                            <1>     ;mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
                            <1>     ; 03/08/2022
00003715 B2C0               <1>     mov     dl, 0C0h
00003717 88D8               <1>     mov     al, bl
00003719 EE                 <1>     out     dx, al
                            <1>     ;mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
                            <1>     ; 03/08/2022
0000371A FEC2               <1>     inc     dl ; dx = 3C1h
0000371C EC                 <1>     in      al, dx
0000371D 8844240D           <1>     mov     [esp+13], al ; bh
                            <1>     ;mov    dx, 3DAh ; VGAREG_ACTL_RESET
                            <1>     ; 03/08/2022
00003721 B2DA               <1>     mov     dl, 0DAh
00003723 EC                 <1>     in      al, dx
                            <1>     ;mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
                            <1>     ; 03/08/2022
00003724 B2C0               <1>     mov     dl, 0C0h
00003726 B020               <1>     mov     al, 20h
00003728 EE                 <1>     out     dx, al
                            <1>     ; ifdef VBOX
                            <1>     ;mov    dx, 3DAh ; VGAREG_ACTL_RESET
                            <1>     ; 03/08/2022
00003729 B2DA               <1>     mov     dl, 0DAh
0000372B EC                 <1>     in      al, dx
                            <1>     ; endif ; VBOX
0000372C E947E4FFFF         <1>     jmp     VIDEO_RETURN
                            <1>
                            <1> set_pel_mask:
                            <1>     ; 10/08/2016
                            <1>     ; BL = mask value
00003731 66BAC603           <1>     mov     dx, 3C6h ; VGAREG_PEL_MASK
00003735 88D8               <1>     mov     al, bl
00003737 EE                 <1>     out     dx, al
00003738 E93BE4FFFF         <1>     jmp     VIDEO_RETURN
                            <1>
                            <1> read_pel_mask:
                            <1>     ; 10/08/2016
                            <1>     ; Output: BL = mask value
0000373D 66BAC603           <1>     mov     dx, 3C6h ; VGAREG_PEL_MASK
00003741 EC                 <1>     in      al, dx
00003742 8844240C           <1>     mov     [esp+12], al ; bl
00003746 E92DE4FFFF         <1>     jmp     VIDEO_RETURN
                            <1>
                            <1> read_single_dac_reg:
                            <1>     ; 02/08/2022
                            <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>
0000374B 66BAC703           <1>     mov     dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
0000374F 88D8               <1>     mov     al, bl
00003751 EE                 <1>     out     dx, al
                            <1>     ;mov    dx, 3C9h ; VGAREG_DAC_DATA
                            <1>     ; 02/08/2022
00003752 B2C9               <1>     mov     dl, 0C9h
00003754 EC                 <1>     in      al, dx
00003755 88442415           <1>     mov     [esp+21], al ; dh
00003759 EC                 <1>     in      al, dx
0000375A 88C5               <1>     mov     ch, al
0000375C EC                 <1>     in      al, dx
0000375D 88C1               <1>     mov     cl, al
0000375F 66894C2410         <1>     mov     [esp+16], cx ; cx
00003764 E90FE4FFFF         <1>     jmp     VIDEO_RETURN
                            <1>
                            <1> read_all_dac_reg:
                            <1>     ; 02/08/2022
                            <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>
00003769 89D7               <1>     mov     edi, edx ; user buffer
0000376B 89CA               <1>     mov     edx, ecx
                            <1>     ;shl    dx, 1 ; *2
                            <1>     ; 02/08/2022
0000376D D1E2               <1>     shl     edx, 1
0000376F 01CA               <1>     add     edx, ecx ; edx = 3*ecx
00003771 89E5               <1>     mov     ebp, esp
00003773 89EE               <1>     mov     esi, ebp
00003775 29D6               <1>     sub     esi, edx
00003777 6683E6FC           <1>     and     si, 0FFFCh ; (dword alignment)
0000377B 89F4               <1>     mov     esp, esi
0000377D 52                 <1>     push    edx ; 3*ecx
0000377E 66BAC703           <1>     mov     dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
00003782 88D8               <1>     mov     al, bl
00003784 EE                 <1>     out     dx, al
00003785 66BAC903           <1>     mov     dx, 3C9h ; VGAREG_DAC_DATA
00003789 89F3               <1>     mov     ebx, esi
                            <1> read_dac_loop:
0000378B EC                 <1>     in      al, dx
0000378C 8803               <1>     mov     [ebx], al
0000378E 43                 <1>     inc     ebx
0000378F EC                 <1>     in      al, dx
00003790 8803               <1>     mov     [ebx], al
00003792 43                 <1>     inc     ebx
00003793 EC                 <1>     in      al, dx
```

```
8479 00003794 8803              <1>     mov     [ebx], al
8480 00003796 43                <1>     inc     ebx
8481                            <1>     ;dec    cx
8482                            <1>     ; 02/08/2022
8483 00003797 49                <1>     dec     ecx
8484 00003798 75F1              <1>     jnz     short read_dac_loop
8485 0000379A 59                <1>     pop     ecx ; 3*ecx
8486                            <1>     ; ECX = transfer (byte) count
8487                            <1>     ; ESI = source address in system space
8488                            <1>     ; EDI = user's buffer address
8489 0000379B E84AD30000        <1>     call    transfer_to_user_buffer
8490 000037A0 89EC              <1>     mov     esp, ebp
8491 000037A2 E9D1E3FFFF        <1>     jmp     VIDEO_RETURN
8492                            <1>
8493                            <1> set_single_dac_reg:
8494                            <1>     ; 03/08/2022 (TRDOS 386 v2.0.5)
8495                            <1>     ; 12/04/2021 (TRDOS 386 v2.0.3, 32 bit push/pop)
8496                            <1>     ; 10/08/2016
8497                            <1>     ; Set One DAC Color Register
8498                            <1>     ; BX = color register to set (0-255)
8499                            <1>        ; CH = green value (00h-3Fh)
8500                            <1>        ; CL = blue value  (00h-3Fh)
8501                            <1>        ; DH = red value   (00h-3Fh)
8502                            <1>
8503                            <1>     ;push   dx
8504                            <1>     ; 12/04/2021
8505 000037A7 52                <1>     push    edx
8506 000037A8 66BAC803          <1>     mov     dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
8507 000037AC 88D8              <1>     mov     al, bl
8508 000037AE EE                <1>     out     dx, al
8509                            <1>     ;;mov   dx, 3C9h ; VGAREG_DAC_DATA
8510                            <1>     ;inc    dx
8511                            <1>     ; 03/08/2022
8512 000037AF FEC2              <1>     inc     dl
8513                            <1>     ;pop    ax
8514                            <1>     ; 12/04/2021
8515 000037B1 58                <1>     pop     eax
8516 000037B2 88E0              <1>     mov     al, ah
8517 000037B4 EE                <1>     out     dx, al
8518 000037B5 88E8              <1>     mov     al, ch
8519 000037B7 EE                <1>     out     dx, al
8520 000037B8 88C8              <1>     mov     al, cl
8521 000037BA EE                <1>     out     dx, al
8522 000037BB E9B8E3FFFF        <1>     jmp     VIDEO_RETURN
8523                            <1>
8524                            <1> set_all_dac_reg:
8525                            <1>     ; 02/08/2022
8526                            <1>     ; 12/08/2016
8527                            <1>     ; 11/08/2016
8528                            <1>     ; 10/08/2016
8529                            <1>     ; Set a Block of DAC Color Register
8530                            <1>     ; BX = first DAC register to set (0-00FFh)
8531                            <1>     ; ECX = number of registers to set (0-00FFh)
8532                            <1>     ; EDX = addr of a table of R,G,B values
8533                            <1>     ;       (it will be CX*3 bytes long)
8534                            <1>
8535 000037C0 89D6              <1>     mov     esi, edx ; user buffer
8536 000037C2 89CA              <1>     mov     edx, ecx
8537                            <1>     ;shl    cx, 1 ; *2
8538                            <1>     ; 02/08/2022
8539 000037C4 D1E1              <1>     shl     ecx, 1
8540 000037C6 01D1              <1>     add     ecx, edx ; ecx = 3*ecx
8541 000037C8 89E5              <1>     mov     ebp, esp
8542 000037CA 89EF              <1>     mov     edi, ebp
8543 000037CC 29CF              <1>     sub     edi, ecx
8544 000037CE 6683E7FC          <1>     and     di, 0FFFCh ; (dword alignment)
8545 000037D2 89FC              <1>     mov     esp, edi
8546 000037D4 E85BD30000        <1>     call    transfer_from_user_buffer
8547                            <1>     ;jc     VIDEO_RETURN
8548                            <1>
8549 000037D9 89D1              <1>     mov     ecx, edx
8550 000037DB 66BAC803          <1>     mov     dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
8551 000037DF 88D8              <1>     mov     al, bl
8552 000037E1 EE                <1>     out     dx, al
8553                            <1>     ;mov    dx, 3C9h ; VGAREG_DAC_DATA
8554                            <1>     ; 02/08/2022
8555 000037E2 FEC2              <1>     inc     dl
8556                            <1> set_dac_loop:
8557 000037E4 8A07              <1>     mov     al, [edi]
8558 000037E6 EE                <1>     out     dx, al
8559 000037E7 47                <1>     inc     edi
8560 000037E8 8A07              <1>     mov     al, [edi]
8561 000037EA EE                <1>     out     dx, al
8562 000037EB 47                <1>     inc     edi
8563 000037EC 8A07              <1>     mov     al, [edi]
8564 000037EE EE                <1>     out     dx, al
8565 000037EF 47                <1>     inc     edi
8566                            <1>     ;dec    cx
8567                            <1>     ; 02/08/2022
8568 000037F0 49                <1>     dec     ecx
8569 000037F1 75F1              <1>     jnz     short set_dac_loop
8570 000037F3 89EC              <1>     mov     esp, ebp
8571 000037F5 E97EE3FFFF        <1>     jmp     VIDEO_RETURN
8572                            <1>
8573                            <1> get_all_palette_reg:
8574                            <1>     ; 03/08/2022
8575                            <1>     ; 10/08/2016
8576                            <1>     ; Read All Palette Registers
8577                            <1>     ; EDX = Address of 17-byte buffer
8578                            <1>     ;       to receive data
8579                            <1>
8580 000037FA 89D7              <1>     mov     edi, edx
8581 000037FC 89E3              <1>     mov     ebx, esp
8582 000037FE 89DE              <1>     mov     esi, ebx
8583 00003800 83EC14            <1>     sub     esp, 20
8584                            <1>
8585 00003803 B100              <1>     mov     cl, 0
8586                            <1> get_palette_loop:
8587 00003805 66BADA03          <1>     mov     dx, 3DAh ; VGAREG_ACTL_RESET
8588 00003809 EC                <1>     in      al, dx
8589                            <1>     ;mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
8590                            <1>     ; 03/08/2022
8591 0000380A B2C0              <1>     mov     dl, 0C0h
8592 0000380C 88C8              <1>     mov     al, cl
8593 0000380E EE                <1>     out     dx, al
8594                            <1>     ;mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
8595                            <1>     ; 03/08/2022
8596                            <1>     ;mov    dl, 0C1h
8597 0000380F FEC2              <1>     inc     dl
8598 00003811 EC                <1>     in      al, dx
8599 00003812 8803              <1>     mov     [ebx], al
8600 00003814 43                <1>     inc     ebx
8601 00003815 FEC1              <1>     inc     cl
8602 00003817 80F910            <1>     cmp     cl, 10h
```

```
8603 0000381A 75E9              <1>     jne     short get_palette_loop
8604                            <1>     ;mov    dx, 3DAh ; VGAREG_ACTL_RESET
8605                            <1>     ; 03/08/2022
8606 0000381C B2DA              <1>     mov     dl, 0DAh
8607 0000381E EC                <1>     in      al, dx
8608                            <1>     ;mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
8609                            <1>     ; 03/08/2022
8610 0000381F B2C0              <1>     mov     dl, 0C0h
8611 00003821 B011              <1>     mov     al, 11h
8612 00003823 EE                <1>     out     dx, al
8613                            <1>     ;mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
8614                            <1>     ; 03/08/2022
8615 00003824 FEC2              <1>     inc     dl ; dx = 3C1h
8616 00003826 EC                <1>     in      al, dx
8617 00003827 8803              <1>     mov     [ebx], al
8618                            <1>     ;mov    dx, 3DAh ; VGAREG_ACTL_RESET
8619                            <1>     ; 03/08/2022
8620 00003829 B2DA              <1>     mov     dl, 0DAh
8621 0000382B EC                <1>     in      al, dx
8622                            <1>     ;mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
8623                            <1>     ; 03/08/2022
8624 0000382C B2C0              <1>     mov     dl, 0C0h
8625 0000382E B020              <1>     mov     al, 20h
8626 00003830 EE                <1>     out     dx, al
8627                            <1>     ; ifdef VBOX
8628                            <1>     ;mov    dx, 3DAh ; VGAREG_ACTL_RESET
8629                            <1>     ; 03/08/2022
8630 00003831 B2DA              <1>     mov     dl, 0DAh
8631 00003833 EC                <1>     in      al, dx
8632                            <1>     ; endif ; VBOX
8633                            <1>
8634                            <1>     ;mov    ecx, 17 ; transfer (byte) count
8635                            <1>     ; 03/08/2022
8636 00003834 29C9              <1>     sub     ecx, ecx
8637 00003836 B111              <1>     mov     cl, 17
8638                            <1>
8639                            <1>     ; ESI = source address in system space
8640                            <1>     ; EDI = user's buffer address
8641 00003838 E8ADD20000        <1>     call    transfer_to_user_buffer
8642                            <1>
8643 0000383D 83C414            <1>     add     esp, 20
8644 00003840 E933E3FFFF        <1>     jmp     VIDEO_RETURN
8645                            <1>
8646                            <1> select_video_dac_color_page:
8647                            <1>     ; 02/08/2022 (TRDOS 386 v2.0.5, code optimization)
8648                            <1>     ; 12/04/2021 (TRDOS 386 v2.0.3, 32 bit push/pop)
8649                            <1>     ; 10/08/2016
8650                            <1>     ; DAC Color Paging Functions
8651                            <1>     ; BL = 00H = select color paging mode
8652                            <1>         ;       BH = paging mode
8653                            <1>         ;            00h = 4 blocks of 64 registers
8654                            <1>         ;            01h = 16 blocks of 16 registers
8655                            <1>     ; BL = 01H = activate color page
8656                            <1>         ;       BH = DAC color page number
8657                            <1>         ;            00h-03h (4-page/64-reg mode)
8658                            <1>         ;            00h-0Fh (16-page/16-reg mode)
8659                            <1>
8660 00003845 66BADA03          <1>     mov     dx, 3DAh ; VGAREG_ACTL_RESET
8661 00003849 EC                <1>     in      al, dx
8662                            <1>     ;mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
8663                            <1>     ; 02/08/2022
8664 0000384A B240              <1>     mov     dl, 40h
8665 0000384C B010              <1>     mov     al, 10h
8666 0000384E EE                <1>     out     dx, al
8667                            <1>     ;mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
8668                            <1>     ; 02/08/2022
8669 0000384F FEC2              <1>     inc     dl ; mov dl, 0C1h
8670 00003851 EC                <1>     in      al, dx
8671 00003852 80E301            <1>     and     bl, 01h
8672 00003855 750C              <1>     jnz     short set_dac_page
8673 00003857 247F              <1>     and     al, 07Fh
8674 00003859 C0E707            <1>     shl     bh, 7
8675 0000385C 08F8              <1>     or      al, bh
8676                            <1>     ;mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
8677                            <1>     ; 02/08/2022
8678 0000385E FECA              <1>     dec     dl ; mov dl, 0C0h
8679 00003860 EE                <1>     out     dx, al
8680 00003861 EB19              <1>     jmp     short set_actl_normal
8681                            <1> set_dac_page:
8682                            <1>     ;push   ax
8683                            <1>     ; 12/04/2021
8684 00003863 50                <1>     push    eax
8685 00003864 66BADA03          <1>     mov     dx, 3DAh ; VGAREG_ACTL_RESET
8686 00003868 EC                <1>     in      al, dx
8687                            <1>     ;mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
8688                            <1>     ; 02/08/2022
8689 00003869 B2C0              <1>     mov     dl, 0C0h
8690 0000386B B014              <1>     mov     al, 14h
8691 0000386D EE                <1>     out     dx, al
8692                            <1>     ;pop    ax
8693                            <1>     ; 12/04/2021
8694 0000386E 58                <1>     pop     eax
8695 0000386F 2480              <1>     and     al, 80h
8696 00003871 7503              <1>     jnz     short set_dac_16_page
8697 00003873 C0E702            <1>     shl     bh, 2
8698                            <1> set_dac_16_page:
8699 00003876 80E70F            <1>     and     bh, 0Fh
8700 00003879 88F8              <1>     mov     al, bh
8701 0000387B EE                <1>     out     dx, al
8702                            <1> set_actl_normal:
8703 0000387C B020              <1>     mov     al, 20h
8704 0000387E EE                <1>     out     dx, al
8705                            <1>     ; ifdef VBOX
8706                            <1>     ;mov    dx, 3DAh ; VGAREG_ACTL_RESET
8707                            <1>     ; 02/08/2022
8708 0000387F B2DA              <1>     mov     dl, 0DAh
8709 00003881 EC                <1>     in      al, dx
8710                            <1>     ; endif ; VBOX
8711 00003882 E9F1E2FFFF        <1>     jmp     VIDEO_RETURN
8712                            <1>
8713                            <1> read_video_dac_state:
8714                            <1>     ; 10/08/2016
8715                            <1>     ; Query DAC Color Paging State
8716                            <1>     ; Output:
8717                            <1>     ; BH = current active DAC color page
8718                            <1>         ; BL = current active DAC paging mode
8719                            <1>
8720 00003887 66BADA03          <1>     mov     dx, 3DAh ; VGAREG_ACTL_RESET
8721 0000388B EC                <1>     in      al, dx
8722 0000388C 66BAC003          <1>     mov     dx, 3C0h ; VGAREG_ACTL_ADDRESS
8723 00003890 B010              <1>     mov     al, 10h
8724 00003892 EE                <1>     out     dx, al
8725 00003893 66BAC103          <1>     mov     dx, 3C1h ; VGAREG_ACTL_READ_DATA
8726 00003897 EC                <1>     in      al, dx
```

```
8727 00003898 88C3              <1>     mov     bl, al
8728 0000389A C0EB07            <1>     shr     bl, 7
8729 0000389D 66BADA03          <1>     mov     dx, 3DAh ; VGAREG_ACTL_RESET
8730 000038A1 EC                <1>     in      al, dx
8731 000038A2 66BAC003          <1>     mov     dx, 3C0h ; VGAREG_ACTL_ADDRESS
8732 000038A6 B014              <1>     mov     al, 14h
8733 000038A8 EE                <1>     out     dx, al
8734 000038A9 66BAC103          <1>     mov     dx, 3C1h ; VGAREG_ACTL_READ_DATA
8735 000038AD EC                <1>     in      al, dx
8736 000038AE 88C7              <1>     mov     bh, al
8737 000038B0 80E70F            <1>     and     bh, 0Fh
8738 000038B3 F6C301            <1>     test    bl, 01
8739 000038B6 7503              <1>     jnz     short get_dac_16_page
8740 000038B8 C0EF02            <1>     shr     bh, 2
8741                            <1> get_dac_16_page:
8742 000038BB 66BADA03          <1>     mov     dx, 3DAh ; VGAREG_ACTL_RESET
8743 000038BF EC                <1>     in      al, dx
8744 000038C0 66BAC003          <1>     mov     dx, 3C0h ; VGAREG_ACTL_ADDRESS
8745 000038C4 B020              <1>     mov     al, 20h
8746 000038C6 EE                <1>     out     dx, al
8747                            <1>     ; ifdef VBOX
8748 000038C7 66BADA03          <1>     mov     dx, 3DAh ; VGAREG_ACTL_RESET
8749 000038CB EC                <1>     in      al, dx
8750                            <1>     ; endif ; VBOX
8751 000038CC 66895C240C        <1>     mov     [esp+12], bx ; bx
8752 000038D1 E9A2E2FFFF        <1>     jmp     VIDEO_RETURN
8753                            <1>
8754                            <1> ; 23/11/2020 - TRDOS 386 v2.0.3
8755                            <1> ; VBE 2 BOCHS/QEMU emulator extensions
8756                            <1> ;   for TRDOS 386 v2 kernel (video bios)
8757                            <1>
8758                            <1> ; BOCH/QEMU VBE2 VGA BIOS code
8759                            <1> ;   by Jeroen Janssen (2002)
8760                            <1> ;   by Volker Rupper (2003-2020)
8761                            <1> ; vbe.c (02/01/2020)
8762                            <1>
8763                            <1> ; vbe.h (02/01/2020)
8764                            <1>
8765                            <1> VBE_DISPI_BANK_ADDRESS    equ    0A0000h
8766                            <1> VBE_DISPI_BANK_SIZE_KB    equ    64
8767                            <1>
8768                            <1> VBE_DISPI_MAX_XRES equ    2560
8769                            <1> VBE_DISPI_MAX_YRES equ    1600
8770                            <1>
8771                            <1> VBE_DISPI_IOPORT_INDEX    equ    01CEh
8772                            <1> VBE_DISPI_IOPORT_DATA     equ    01CFh
8773                            <1>
8774                            <1> VBE_DISPI_INDEX_ID equ    00h
8775                            <1> VBE_DISPI_INDEX_XRES      equ    01h
8776                            <1> VBE_DISPI_INDEX_YRES      equ    02h
8777                            <1> VBE_DISPI_INDEX_BPP       equ    03h
8778                            <1> VBE_DISPI_INDEX_ENABLE    equ    04h
8779                            <1> VBE_DISPI_INDEX_BANK      equ    05h
8780                            <1> VBE_DISPI_INDEX_VIRT_WIDTH equ   06h
8781                            <1> VBE_DISPI_INDEX_VIRT_HEIGHT equ  07h
8782                            <1> VBE_DISPI_INDEX_X_OFFSET equ     08h
8783                            <1> VBE_DISPI_INDEX_Y_OFFSET equ     09h
8784                            <1> VBE_DISPI_INDEX_VIDEO_MEMORY_64K equ 0Ah
8785                            <1> VBE_DISPI_INDEX_DDC       equ    0Bh
8786                            <1>
8787                            <1> VBE_DISPI_ID0             equ    0B0C0h
8788                            <1> VBE_DISPI_ID1             equ    0B0C1h
8789                            <1> VBE_DISPI_ID2             equ    0B0C2h
8790                            <1> VBE_DISPI_ID3             equ    0B0C3h
8791                            <1> VBE_DISPI_ID4             equ    0B0C4h
8792                            <1> VBE_DISPI_ID5             equ    0B0C5h
8793                            <1>
8794                            <1> VBE_DISPI_DISABLED equ    00h
8795                            <1> VBE_DISPI_ENABLED  equ    01h
8796                            <1> VBE_DISPI_GETCAPS  equ    02h
8797                            <1> VBE_DISPI_8BIT_DAC equ    20h
8798                            <1> VBE_DISPI_LFB_ENABLED     equ    40h
8799                            <1> VBE_DISPI_NOCLEARMEM      equ    80h
8800                            <1>
8801                            <1> VBE_DISPI_LFB_PHYSICAL_ADDRESS equ 0E0000000h
8802                            <1>
8803                            <1> ; ***
8804                            <1>
8805                            <1> ;// VBE Return Status Info
8806                            <1> ;// AL
8807                            <1> VBE_RETURN_STATUS_SUPPORTED       equ    4Fh
8808                            <1> VBE_RETURN_STATUS_UNSUPPORTED     equ    00h
8809                            <1> ;// AH
8810                            <1> VBE_RETURN_STATUS_SUCCESSFULL     equ    00h
8811                            <1> VBE_RETURN_STATUS_FAILED  equ    01h
8812                            <1> VBE_RETURN_STATUS_NOT_SUPPORTED   equ    02h
8813                            <1> VBE_RETURN_STATUS_INVALID equ    03h
8814                            <1>
8815                            <1> ;// VBE Mode Numbers
8816                            <1>
8817                            <1> VBE_MODE_VESA_DEFINED             equ    0100h
8818                            <1> VBE_MODE_REFRESH_RATE_USE_CRTC    equ    0800h
8819                            <1> VBE_MODE_LINEAR_FRAME_BUFFER      equ    4000h
8820                            <1> VBE_MODE_PRESERVE_DISPLAY_MEMORY equ     8000h
8821                            <1>
8822                            <1> ;// Mode Attributes
8823                            <1>
8824                            <1> VBE_MODE_ATTRIBUTE_SUPPORTED                equ 0001h
8825                            <1> VBE_MODE_ATTRIBUTE_EXTENDED_INFO_AVAILABLE equ  0002h
8826                            <1> VBE_MODE_ATTRIBUTE_COLOR_MODE               equ 0008h
8827                            <1> VBE_MODE_ATTRIBUTE_GRAPHICS_MODE     equ 0010h
8828                            <1> VBE_MODE_ATTRIBUTE_LINEAR_FRAME_BUFFER_MODE equ 0080h
8829                            <1> VBE_MODE_ATTRIBUTE_DOUBLE_SCAN_MODE         equ 0100h
8830                            <1> VBE_MODE_ATTRIBUTE_INTERLACE_MODE    equ 0200h
8831                            <1>
8832                            <1> ;// Window attributes
8833                            <1>
8834                            <1> VBE_WINDOW_ATTRIBUTE_RELOCATABLE equ     01h
8835                            <1> VBE_WINDOW_ATTRIBUTE_READABLE      equ   02h
8836                            <1> VBE_WINDOW_ATTRIBUTE_WRITEABLE     equ   04h
8837                            <1>
8838                            <1> ;/* Video memory */
8839                            <1> VGAMEM_GRAPH equ 0A000h
8840                            <1> VGAMEM_CTEXT equ 0B800h
8841                            <1> ;VGAMEM_MTEXT equ 0B000h
8842                            <1>
8843                            <1> ;// Memory model
8844                            <1>
8845                            <1> ;VBE_MEMORYMODEL_TEXT_MODE equ     00h
8846                            <1> ;VBE_MEMORYMODEL_CGA_GRAPHICS     equ    01h
8847                            <1> ;VBE_MEMORYMODEL_PLANAR           equ    03h
8848                            <1> VBE_MEMORYMODEL_PACKED_PIXEL      equ    04h
8849                            <1> ;VBE_MEMORYMODEL_NON_CHAIN_4_256 equ     05h
8850                            <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>
 000038D6 55            <1>     push    ebp ; ***
 000038D7 56            <1>     push    esi ; ****
                        <1>
 000038D8 31FF          <1>     xor     edi, edi  ; mov edi, 0
                        <1>
 000038DA 803D[86090000]03      <1>     cmp     byte [vbe3], 3
 000038E1 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>
 000038E3 66B8014F      <1>     mov     ax, 4F01h
                        <1>
 000038E7 E826E0FFFF    <1>     call    _vbe3_pmfn_return_mode_info
                        <1>
 000038EC 6683F84F      <1>     cmp     ax, 004Fh
 000038F0 7533          <1>     jne     short _vbe_rmi_2 ; fail
                        <1>
                        <1>     ; 15/12/2020
                        <1>     ; cx = vbe video mode
 000038F2 80E501        <1>     and     ch, 01h ; clear LFB flag
 000038F5 BEFE7B0900    <1>     mov     esi, VBE3MODEINFOBLOCK - 2
 000038FA 66890E        <1>     mov     [esi], cx ; MODEINFO.mode
 000038FD E8A5000000    <1>     call    set_lfbinfo_table
 00003902 EB22          <1>     jmp     short _vbe_rmi_3  ; cf = 0
                        <1> _vbe_rmi_1:
 00003904 803D[86090000]02      <1>     cmp     byte [vbe3], 2
 0000390B 7219          <1>     jb      short _vbe_rmi_3 ; cf = 1
 0000390D A0[87090000]  <1>     mov     al, [vbe2bios] ; 0C0h-0C5h for emu (*)
 00003912 3CC0          <1>     cmp     al, 0C0h ; BOCHS/QEMU/VIRTUALBOX (*) ?
 00003914 7210          <1>     jb      short _vbe_rmi_3  ; cf = 1
 00003916 3CC5          <1>     cmp     al, 0C5h ; (*)
 00003918 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>
 0000391A E80A000000    <1>     call    vbe_biosfn_return_mode_info
 0000391F 6683F84F      <1>     cmp     ax, 004Fh ; successful ?
 00003923 7401          <1>     je      short _vbe_rmi_3  ; cf = 0
                        <1> _vbe_rmi_2:
 00003925 F9            <1>     stc
                        <1>     ; cf = 1
                        <1> _vbe_rmi_3:
 00003926 5E            <1>     pop     esi ; ****
 00003927 5D            <1>     pop     ebp ; ***
                        <1>
                        <1>     ;pop    es  ; **
                        <1>     ;pop    ss  ; *
                        <1>
 00003928 C3            <1>     retn
                        <1>
                        <1>
                        <1> ; * (TRDOS 386, INT 31h, VESA Video Bios functions)
                        <1> ; * ------------------------------------------------------
                        <1> ; * Function 01h - Return VBE Mode Information
                        <1> ; * ------------------------------------------------------
                        <1> ; * Input:
                        <1> ; *         AX = 4F01h
                        <1> ; *         CX = Mode number
                        <1> ; *    (ES:DI) EDI = Pointer to ModeInfoBlock structure
                        <1> ; * Output:
                        <1> ; *         AX = VBE Return Status
                        <1> ; *
                        <1> ; *------------------------------------------------------
                        <1> ; *
                        <1>
                        <1> vbe_biosfn_return_mode_info:
                        <1>     ; 03/08/2022 (TRDOS 386 v2.0.5)
                        <1>     ; 15/12/2020
                        <1>     ; 14/12/2020
                        <1>     ; 12/12/2020
                        <1>     ; 11/12/2020 (TRDOS 386 v2.0.3)
                        <1>     ;
                        <1>     ; Input:
                        <1>     ;       cx = video (bios) mode
                        <1>     ;      edi = ModeInfoBlock buffer address
                        <1>     ;            (in user's memory space)
                        <1>     ;      (ax = 4F01h)
                        <1>     ; Output:
                        <1>     ;       ax = 004Fh (successful)
                        <1>     ;       ah > 0 -> error
                        <1>     ;
```

```
                                <1>     ; Modified registers: esi
                                <1>
                                <1>     ;;push ds  ; *
                                <1>     ;;push es  ; **
                                <1>     ;;push ebp ; ***
                                <1>     ;;push esi ; ****
                                <1>
00003929 F6C501                 <1>     test    ch, 1
0000392C 7505                   <1>     jnz     short vbe_rmi_1
                                <1>
                                <1>     ; mode number < 100h
                                <1>     ; CGA/VGA mode is not proper this VBE function
                                <1>
0000392E 29C0                   <1>     sub     eax, eax
                                <1> vbe_rmi_0:
                                <1>     ;mov    ax, 0100h  ; Function is not supported
00003930 B401                   <1>     mov     ah, 1
00003932 C3                     <1>     retn
                                <1> vbe_rmi_1:
00003933 52                     <1>     push    edx ; *****
00003934 51                     <1>     push    ecx ; ******
00003935 53                     <1>     push    ebx ; *******
00003936 57                     <1>     push    edi ; ********
                                <1>
                                <1>     ; 14/12/2020
00003937 89CB                   <1>     mov     ebx, ecx
                                <1>
                                <1>     ;xor    eax, eax
00003939 80E7C1                 <1>     and     bh, 0C1h ; use bit 15, 14, 8 only (for bh)
0000393C 883D[ED9C0100]         <1>     mov     [vbe_mode_x], bh
                                <1>     ;and    bx, 1FFh
00003942 80E701                 <1>     and     bh, 1
                                <1>     ;mov    bh, 1
                                <1>
                                <1>     ; Alternative 2 (instead of 'Mode_info_find_mode')
00003945 E86A060000             <1>     call    set_mode_info_list ; (alternative 2)
                                <1>
                                <1>     ; eax = 0
                                <1>
                                <1>     ;mov    bx, [esi] ; mode
                                <1>
                                <1>     ; Alternative 1 (instead of 'set_mode_info_list')
                                <1>     ;call   mode_info_find_mode ; (alternative 1)
                                <1>
0000394A 09F6                   <1>     or      esi, esi
                                <1>     ; 14/12/2020
0000394C 744C                   <1>     jz      short vbe_rmi_4       ; VBE mode number is wrong
                                <1>                                   ; or it is not supported
                                <1>
                                <1>     ; 15/12/2020
                                <1>     ;mov    bx, [esi] ; mode
                                <1>
                                <1>     ; 12/12/2020
                                <1>     ;call   set_lfbinfo_table
                                <1>
0000394E F605[ED9C0100]40       <1>     test    byte [vbe_mode_x], 40h ; LFB model ?
00003955 7404                   <1>     jz      short vbe_rmi_2
                                <1>
00003957 C6461C01               <1>     mov     byte [esi+MODEINFO.NumberOfBanks], 1
                                <1> vbe_rmi_2:
                                <1>     ; (vbe.c, 02/01/2020, vruppert)
                                <1>     ; 11/12/2020 (Erdogan Tan, video.s)
                                <1>     ; Bochs Graphics Adapter
                                <1>     ; vendor_id: 1111h, device id: 1234h
                                <1>
0000395B E835070000             <1>     call    pci_get_lfb_addr
                                <1>     ;or     eax, eax
00003960 7404                   <1>     jz      short vbe_rmi_3
                                <1>     ; zf = 0, ax > 0 (high word of LFB address)
                                <1>     ; set/change LFB address in MODEINFO structure
00003962 6689462C               <1>     mov     [esi+MODEINFO.PhysBasePtr+2], ax
                                <1>     ; 12/12/2020
                                <1>     ;mov    [edi+LFBINFO.LFB_addr+2], ax
                                <1> vbe_rmi_3:
                                <1>     ;test   byte [esi+MODEINFO.WinAAttributes], 1
                                <1>     ;               ; VBE_WINDOW_ATTRIBUTE_RELOCATABLE = 1
                                <1>         ;jz short vbe_rmi_4
                                <1>     ;; 11/12/2020
                                <1>     ;; In fact, this is far call address in (Bochs/BGA) Video Bios
                                <1>     ;; Direct user access to kernel subroutines is not possible
                                <1>     ;; in TRDOS 386. Also, TRDOS 386 kernel will support only LFB.
                                <1>     ;; Bank select may be a seperate sysvideo function in future
                                <1>     ;; (if it will be required).
                                <1>     ;mov    dword [esi+MODEINFO.WinFuncPtr], dispi_set_bank_farcall
                                <1> ;vbe_rmi_4:
                                <1>     ; 12/12/2020
00003966 E83C000000             <1>     call    set_lfbinfo_table
                                <1>
                                <1>     ; 11/12/2020
                                <1>     ; copy 68 bytes of MODE_INFO_LIST to user
                                <1>
0000396B 8B3C24                 <1>     mov     edi, [esp]  ; user's buffer address
                                <1>     ; 12/12/2020
0000396E 09FF                   <1>     or      edi, edi ; 0 = kernel call
                                <1>                      ; (call from '_vbe_biosfn_return_mode_info')
00003970 7431                   <1>     jz      short vbe_rmi_6
                                <1>
                                <1>     ; 15/12/2020
                                <1>     ; prepare 256 bytes MODEINFO buffer at VBE3MODEINFOBLOCK
                                <1>     ; and then, copy buffer conttent to user's buffer
00003972 57                     <1>     push    edi
00003973 BE[0C9D0100]           <1>     mov     esi, MODE_INFO_LIST + 2       ; MODEINFO.ModeAttributes
00003978 BF007C0900             <1>     mov     edi, VBE3MODEINFOBLOCK
                                <1>     ;mov    ecx, 66/4  ; 66 bytes
                                <1>     ; 03/08/2022
0000397D 29C9                   <1>     sub     ecx, ecx
0000397F B110                   <1>     mov     cl, 66/4
00003981 F3A5                   <1>     rep     movsd
00003983 31C0                   <1>     xor     eax, eax
00003985 B12F                   <1>     mov     cl, (256-68)/4 ; 188 bytes
00003987 F3AB                   <1>     rep     stosd
00003989 66AB                   <1>     stosw  ; 2 bytes
0000398B 5F                     <1>     pop     edi
0000398C BE007C0900             <1>     mov     esi, VBE3MODEINFOBLOCK
                                <1>     ;mov    cx, 256
00003991 FEC5                   <1>     inc     ch ; cx = 256
00003993 E852D10000             <1>     call    transfer_to_user_buffer
00003998 7309                   <1>     jnc     short vbe_rmi_5
                                <1> vbe_rmi_4:
                                <1>     ;mov    eax, 014Fh ; fail/error
0000399A 31C0                   <1>     xor     eax, eax
0000399C B401                   <1>     mov     ah, 01h
                                <1>     ;jmp    short vbe_rmi_6
0000399E E981000000             <1>     jmp     vbe_sm_ret1 ; 11/12/2020
```

```
                                <1> vbe_rmi_5:
                                <1>     ; 256 bytes of MODEINFO have been transferred to user
                                <1>     ;mov    eax, 4Fh ; succesfull
                                <1> vbe_rmi_6: ; 12/12/2020
000039A3 31C0                   <1>     xor     eax, eax
                                <1> ;vbe_rmi_6:
000039A5 EB7D                   <1>     jmp     vbe_sm_ret1 ; 11/12/2020
                                <1>
                                <1>     ;pop    edi ; ********
                                <1>     ;pop    ebx ; *******
                                <1>     ;pop    ecx ; ******
                                <1>     ;pop    edx ; *****
                                <1>
                                <1>     ;;pop   esi ; ****
                                <1>     ;;pop   ebp ; ***
                                <1>     ;;pop   es  ; **
                                <1>     ;;pop   ds  ; *
                                <1>
                                <1>     ;retn
                                <1>
                                <1> set_lfbinfo_table:
                                <1>     ; 19/12/2020
                                <1>     ; 11/12/2020
                                <1>     ; Set/Fill LFBINFO structure/table
                                <1>     ;
                                <1>     ; Input:
                                <1>     ;       esi = Mode info list address
                                <1>     ; Output:
                                <1>     ;       LFB_Info address is filled with LFBINFO
                                <1>     ;       edi = LFB_Info address
                                <1>     ;
                                <1>     ; Modified registers: eax, edx (=0), edi
                                <1>
000039A7 BF[FA9C0100]           <1>     mov     edi, LFB_Info
000039AC 8B462A                 <1>     mov     eax, [esi+MODEINFO.PhysBasePtr]
000039AF 894702                 <1>     mov     [edi+LFBINFO.LFB_addr], eax ; LFB address
                                <1>     ;mov    ax, [esi+MODEINFO.mode]
000039B2 668B06                 <1>     mov     ax, [esi]
000039B5 668907                 <1>     mov     [edi+LFBINFO.mode],ax
000039B8 8A461B                 <1>     mov     al, [esi+MODEINFO.BitsPerPixel]
000039BB 88470E                 <1>     mov     [edi+LFBINFO.bpp], al
000039BE 29C0                   <1>     sub     eax, eax
000039C0 668B4614               <1>     mov     ax, [esi+MODEINFO.XResolution]
000039C4 6689470A               <1>     mov     [edi+LFBINFO.X_res], ax
000039C8 89C2                   <1>     mov     edx, eax ; 19/12/2020
000039CA 668B4616               <1>     mov     ax, [esi+MODEINFO.YResolution]
000039CE 6689470C               <1>     mov     [edi+LFBINFO.Y_res], ax
                                <1>     ; eax = Y_res ; screen height
                                <1>     ; 19/12/2020
000039D2 F7E2                   <1>     mul     edx ; X_res*Y_res
                                <1>     ; edx = 0
000039D4 8A570E                 <1>     mov     dl, [edi+LFBINFO.bpp]
                                <1>     ; Note:
                                <1>     ; Bits per pixel may be 8,16,24,32 for TRDOS 386 v2.
                                <1>     ; (4 bits for pixel is not used for VESA modes here)
000039D7 C0EA03                 <1>     shr     dl, 3 ; convert bits to byte
000039DA F7E2                   <1>     mul     edx
                                <1>     ; eax = screen/page/buffer size in bytes
000039DC 894706                 <1>     mov     [edi+LFBINFO.LFB_size], eax
                                <1>     ; edx = 0
                                <1>     ; clear reserved byte in LFBINFO structure/table
000039DF 88570F                 <1>     mov     [edi+LFBINFO.reserved], dl ; not necessary
000039E2 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>     ; 07/03/2021
                                <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
000039E3 52                     <1>     push    edx ; *****
000039E4 51                     <1>     push    ecx ; ******
000039E5 53                     <1>     push    ebx ; *******
000039E6 57                     <1>     push    edi ; ********
                                <1>
                                <1>     ;xor    eax, eax
000039E7 80E7C1                 <1>     and     bh, 0C1h ; use bit 15, 14, 8 only (for bh)
000039EA 883D[ED9C0100]         <1>     mov     [vbe_mode_x], bh
000039F0 80E701                 <1>     and     bh, 1
000039F3 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
000039F5 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
000039F7 8405[ED9C0100]         <1>     test   [vbe_mode_x], al ; 80h
000039FD 7402                   <1>     jz     short vbe_sm_1 ; clear display memory
                                <1>     ; no_clear
000039FF 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
00003A01 31C0                   <1>     xor    eax, eax ; 0
00003A03 E89C040000             <1>     call   dispi_set_enable
                                <1>
00003A08 88D8                   <1>     mov    al, bl
                                <1>     ;jmp   _set_mode ; (in 'biosfn_set_video_mode' sub)
00003A0A E87AE1FFFF             <1>     call   _set_mode ; will return with cf=1 only if
                                <1>                      ; desired mode is not implemented
                                <1>     ; _retn: return from subroutine
00003A0F 721A                   <1>     jc     short vbe_sm_2 ; 25/11/2020
                                <1>
                                <1>     ; 26/11/2020
00003A11 31C0                   <1>     xor    eax, eax
00003A13 A0[1E680000]           <1>     mov    al, [CRT_MODE]
                                <1>     ; 27/11/2020
00003A18 8A25[17830100]         <1>     mov    ah, [noclearmem] ; 80h or 0
                                <1>     ;and   ah 80h
00003A1E 66A3[EE9C0100]         <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
00003A24 B04F                   <1>     mov    al, 4Fh ; Function call successful
                                <1>     ; eax = 004Fh
                                <1> vbe_sm_ret2:
                                <1>     ; 11/12/2020
00003A26 5F                     <1>     pop    edi ; ********
00003A27 5B                     <1>     pop    ebx ; *******
00003A28 59                     <1>     pop    ecx ; ******
00003A29 5A                     <1>     pop    edx ; *****
                                <1>
                                <1>     ;;pop  esi ; ****
                                <1>     ;;pop  ebp ; ***
                                <1>     ;;pop  es  ; **
                                <1>     ;;pop  ds  ; *
                                <1>
00003A2A C3                     <1>     retn
                                <1>
                                <1> vbe_sm_2:
                                <1>     ;mov   ax, 0100h ; Function is not supported
                                <1>     ; 27/11/2020
00003A2B 31C0                   <1>     xor    eax, eax
00003A2D B401                   <1>     mov    ah, 01h
                                <1>     ; eax = 0100h
                                <1>     ;retn
00003A2F 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
00003A31 803D[1E680000]03       <1>     cmp    byte [CRT_MODE], 03h
                                <1>     ;jne   short vbe_sm_0
00003A38 7505                   <1>     jne    short vbe_sm_4 ; 07/03/2021
00003A3A E8ECE3FFFF             <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
00003A3F E870050000             <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>
00003A44 09F6                   <1>     or     esi, esi
00003A46 74E3                   <1>     jz     short vbe_sm_2 ; VBE mode number is wrong
                                <1>                           ; or it is not supported
                                <1>
```

```
                                <1>     ; 11/12/2020
00003A48 668B1E                 <1>     mov     bx, [esi] ; mode
                                <1>
                                <1>     ; 27/11/2020
00003A4B 0A3D[ED9C0100]         <1>     or      bh, [vbe_mode_x]
                                <1>
                                <1>     ; save VESA VBE mode
00003A51 66891D[EE9C0100]       <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
00003A58 29C0                   <1>     sub     eax, eax ; 0
                                <1>
00003A5A E845040000             <1>     call    dispi_set_enable
                                <1>
                                <1>     ; 11/12/2020
00003A5F 8A461B                 <1>     mov     al, [esi+MODEINFO.BitsPerPixel]
                                <1>     ; ah = 0
                                <1>
                                <1>     ;cmp    byte [esi+MODEINFO.BitsPerPixel], 8
00003A62 3C08                   <1>     cmp     al, 8
00003A64 750B                   <1>     jne     short vbe_sm_5
                                <1>
                                <1>     ; 11/12/2020
                                <1>     ;push   edi
00003A66 50                     <1>     push    eax
                                <1>     ; 'load_dac_palette(3);'
00003A67 56                     <1>     push    esi
00003A68 B403                   <1>     mov     ah, 3  ; palette3, 256 colors
00003A6A E8BBF1FFFF             <1>     call    load_dac_palette
00003A6F 5E                     <1>     pop     esi
                                <1>     ; 11/12/2020
00003A70 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]
00003A71 E841040000             <1>     call    dispi_set_bpp
                                <1>         ;'dispi_set_xres(cur_info->info.XResolution);'
00003A76 668B4614               <1>     mov     ax, [esi+MODEINFO.XResolution]
00003A7A E83E040000             <1>     call    dispi_set_xres
                                <1>         ;'dispi_set_yres(cur_info->info.YResolution);'
00003A7F 668B4616               <1>     mov     ax, [esi+MODEINFO.YResolution]
00003A83 E83B040000             <1>     call    dispi_set_yres
                                <1>
                                <1>     ;'dispi_set_bank(0);'
                                <1>     ;xor    ax, ax
00003A88 31C0                   <1>     xor     eax, eax ; 0
00003A8A E83A040000             <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
00003A8F A0[ED9C0100]           <1>     mov     al, [vbe_mode_x] ; restore VBE mode bit 14 & 15
00003A94 0C01                   <1>     or      al, 1 ; VBE_DISPI_ENABLED
00003A96 E809040000             <1>     call    dispi_set_enable
                                <1>
                                <1>         ; 'vga_compat_setup();'
00003A9B E83E040000             <1>     call    vga_compat_setup
                                <1>
                                <1>     ; 11/12/2020
00003AA0 E802FFFFFF             <1>     call    set_lfbinfo_table
                                <1>
                                <1>     ; 26/11/2020
00003AA5 31C0                   <1>     xor     eax, eax
00003AA7 FEC8                   <1>     dec     al
00003AA9 A2[1E680000]           <1>     mov     [CRT_MODE], al ; 0FFh = VESA VBE mode sign
                                <1>
                                <1>     ; 27/11/2020
00003AAE 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>
00003AB3 A0[1E680000]       <1>     mov     al, [CRT_MODE] ; current cga/vga mode
00003AB8 3CFF               <1>     cmp     al, 0FFh ; VBE extension signature
00003ABA 720E               <1>     jb      short vbe_gm_1 ; get CGA/VGA mode
                            <1>
                            <1>     ; get VBE mode
                            <1> vbe_gm_0:
00003ABC 66A1[EE9C0100]     <1>     mov     ax, [video_mode]
                            <1>             ; BX bits:
                            <1>             ; bit 0 to 8 = VESA VBE video mode
                            <1>             ; bit 9 to 13 = 0
                            <1>             ; bit 14 = last mode set LFB option
                            <1>             ;          1 - linear/flat frame buffer
                            <1>             ;          0 - windowed frame buffer
                            <1>             ; bit 15 = last mode set no_clear option
                            <1>             ;          0 - video memory cleared
                            <1>             ;          1 - video memory not cleared
                            <1>
                            <1> vbe_gm_return:
                            <1>     ;pop    edx ; ******
00003AC2 0FB7D8             <1>     movzx   ebx, ax
                            <1> ;vbe_srs_retn:
00003AC5 31C0               <1>     xor     eax, eax ; 0
00003AC7 B04F               <1>     mov     al, 4Fh ; ax = 004Fh (successful)
00003AC9 C3                 <1>     retn
                            <1>
                            <1> vbe_gm_1:
                            <1>     ; legacy (old, standard) CGA/VGA bios video mode
00003ACA 8A25[17830100]     <1>     mov     ah, [noclearmem] ; 80h or 0
                            <1>             ; BX bits:
                            <1>             ; bit 0 to 7 = video mode
                            <1>             ; bit 8 to 13 = 0
                            <1>             ; bit 14 = 0 (not LFB mode) CGA/VGA
                            <1>             ; bit 15 = 1 if [noclearmem] = 80h
                            <1>             ;          0 if [noclearmem] = 0
00003AD0 EBF0               <1>     jmp     short vbe_gm_return
                            <1>
                            <1> ; * (TRDOS 386, INT 31h, VESA Video Bios functions)
                            <1> ; * ------------------------------------------------------
                            <1> ; * Function 04h - Save/Restore State
                            <1> ; * ------------------------------------------------------
                            <1> ; * Input:
                            <1> ; *         AX = 4F04h
                            <1> ; *              DL = 00h Return Save/Restore State buff size
                            <1> ; *                   01h Save State
                            <1> ; *                   02h Restore State
                            <1> ; *              CX = Requested states
                            <1> ; *               bit 0 - controller hardware state
                            <1> ; *               bit 1 - BIOS data state
                            <1> ; *               bit 2 - DAC state
                            <1> ; *               bit 3 - register state
                            <1> ; *    (ES:BX) EBX = Pointer to buffer (if DL <> 00h)
                            <1> ; * Output:
                            <1> ; *         AX = VBE Return Status
                            <1> ; *         BX = Number of 64-byte blocks
                            <1> ; *              to hold the state buffer (if DL=00h)
                            <1> ; *
                            <1> ; *-------------------------------------------------------
                            <1> ; *
                            <1>
                            <1> vbe_biosfn_save_restore_state:
                            <1>     ; 23/01/2021
                            <1>     ; 16/01/2021
                            <1>     ; 14/01/2021
                            <1>     ; 13/01/2021
                            <1>     ; 12/01/2021
                            <1>     ; 11/01/2021 (TRDOS 386 v2.0.3)
                            <1>     ; (ref: vbe.c, 02/01/2020, vruppert)
                            <1>     ;
                            <1>     ; Input:
                            <1>     ;       dl = sub function
                            <1>     ;       cl = requested state
                            <1>     ;      ebx = pointer to buffer (if dl<>00h)
                            <1>     ; Output:
                            <1>     ;       ax = 004Fh (successful)
                            <1>     ;       ah > 0 -> error
                            <1>     ;       bx = Number of 64-byte blocks
                            <1>     ;            to hold the state buffer (if DL=00h)
                            <1>
                            <1>     ; Modified registers: eax, ebx, edi
                            <1>
                            <1>     ; 14/01/2021
00003AD2 09DB               <1>     or      ebx, ebx ; user's buffer address
00003AD4 750A               <1>     jnz     short _vbe_biosfn_save_restore_state
                            <1>
00003AD6 20D2               <1>     and     dl, dl
00003AD8 7406               <1>     jz      short _vbe_biosfn_save_restore_state
                            <1>
                            <1>     ; function failed
                            <1>     ;mov    eax, 0100h
                            <1>     ;xor    eax, eax
                            <1>     ;inc    ah  ; eax = 0100h
                            <1>     ; 16/01/2021
00003ADA B84F010000         <1>     mov     eax, 014Fh
00003ADF C3                 <1>     retn
                            <1>
                            <1> _vbe_biosfn_save_restore_state:
                            <1>     ; 23/01/2021
                            <1>     ; 14/01/2021
                            <1>     ; ebx = 0 if the caller is kernel ('sysvideo')
                            <1>
                            <1>     ; 13/01/2021
00003AE0 57                 <1>     push    edi
00003AE1 52                 <1>     push    edx
00003AE2 51                 <1>     push    ecx
                            <1>
                            <1>     ; 23/01/2021
                            <1>     ; 12/01/2021
00003AE3 80FA02             <1>     cmp     dl, 2
00003AE6 7757               <1>     ja      short vbe_srs_7 ; 23/01/2021
                            <1>                     ; invalid sub function
00003AE8 83F90F             <1>     cmp     ecx, 0Fh
00003AEB 7752               <1>     ja      short vbe_srs_7 ; invalid !
                            <1>
00003AED 20D2               <1>     and     dl, dl
00003AEF 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]
00003AF1 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:
00003AF6 6683C33F           <1>     add     bx, 63
00003AFA 66C1EB06           <1>     shr     bx, 6 ; / 64
                            <1>
                            <1> vbe_srs_retn:
00003AFE 31C0               <1>     xor     eax, eax ; 0
                            <1> vbe_srs_0:  ; 16/01/2021
00003B00 B04F               <1>     mov     al, 4Fh ; ax = 004Fh (successful)
                            <1> ;vbe_srs_0:
                            <1>     ; 13/01/2021
00003B02 59                 <1>     pop     ecx
00003B03 5A                 <1>     pop     edx
00003B04 5F                 <1>     pop     edi
                            <1>
00003B05 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
00003B06 80E10F             <1>     and     cl, 0Fh ; 8, 4, 2, 1
00003B09 7434               <1>     jz      short vbe_srs_7 ; cx = 0 -> invalid !
                            <1>
00003B0B BF00760900         <1>     mov     edi, VBE3SAVERESTOREBLOCK
                            <1>
00003B10 80FA01             <1>     cmp     dl, 1
00003B13 7730               <1>     ja      short vbe_srs_8
                            <1>
                            <1>     ; save video state
                            <1>
00003B15 F6C107             <1>     test    cl, 07h ; 4, 2, 1
00003B18 740A               <1>     jz      short vbe_srs_5  ; vbe dispi regs state
                            <1>
00003B1A 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
00003B1F F6C108             <1>     test    cl, 8
00003B22 7405               <1>     jz      short vbe_srs_6
                            <1> vbe_srs_5:
00003B24 E8AB010000         <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
00003B29 21DB               <1>     and     ebx, ebx
00003B2B 74D1               <1>     jz      short vbe_srs_retn ; the caller is kernel
                            <1>
00003B2D BE00760900         <1>     mov     esi, VBE3SAVERESTOREBLOCK
00003B32 29F7               <1>     sub     edi, esi
00003B34 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>
00003B36 89DF               <1>     mov     edi, ebx ; user's buffer address
00003B38 E8ADCF0000         <1>     call    transfer_to_user_buffer
00003B3D 73BF               <1>     jnc     short vbe_srs_retn
                            <1> vbe_srs_7:
                            <1>     ; // function failed
                            <1>     ;mov    eax, 0100h
00003B3F 31C0               <1>     xor     eax, eax
00003B41 FEC4               <1>     inc     ah  ; eax = 0100h
                            <1>     ; 16/01/2021
                            <1>     ; ax = 014Fh
                            <1>     ;retn
                            <1>     ; 13/01/2021
00003B43 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
00003B45 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
00003B47 7414               <1>     jz      short vbe_srs_12 ; 'sysvideo' call
                            <1> vbe_srs_11:
00003B49 89DE               <1>     mov     esi, ebx ; user's buffer address
                            <1>     ; 23/01/2021
                            <1>     ;push   ebx
                            <1>
```

```
9719 00003B4B E827000000          <1>     call    vbe_srs_gbs
9720                              <1>
9721                              <1>     ; restore video state
9722                              <1>
9723                              <1>     ;mov    edi, VBE3SAVERESTOREBLOCK
9724 00003B50 51                  <1>     push    ecx
9725 00003B51 89D9                <1>     mov     ecx, ebx ; transfer count in bytes
9726 00003B53 E8DCCF0000          <1>     call    transfer_from_user_buffer
9727 00003B58 59                  <1>     pop     ecx
9728                              <1>     ; 23/01/2021
9729                              <1>     ;pop    ebx
9730 00003B59 89F3                <1>     mov     ebx, esi
9731 00003B5B 72E2                <1>     jc      short vbe_srs_7
9732                              <1>
9733                              <1> vbe_srs_12:
9734                              <1>     ;mov    esi, VBE3SAVERESTOREBLOCK
9735 00003B5D 89FE                <1>     mov     esi, edi
9736                              <1>
9737 00003B5F F6C107              <1>     test    cl, 07h ; 4, 2, 1
9738 00003B62 740C                <1>     jz      short vbe_srs_9       ; vbe dispi regs state
9739                              <1>
9740 00003B64 E8A2010000          <1>     call    biosfn_restore_video_state
9741 00003B69 72D4                <1>     jc      short vbe_srs_7 ; invalid buffer content !
9742                              <1>     ; esi = current position
9743                              <1>     ;       in VBE3SAVERESTOREBLOCK
9744                              <1>     ;       (VGA save_state offset)
9745                              <1>     ; modified regs: esi, eax, edx, ch
9746 00003B6B F6C108              <1>     test    cl, 8
9747                              <1>     ;jz     short vbe_srs_10
9748                              <1>     ; 23/01/2020
9749 00003B6E EB8E                <1>     jmp     short vbe_srs_retn
9750                              <1> vbe_srs_9:
9751 00003B70 E8F1020000          <1>     call    vbe_biosfn_restore_video_state
9752                              <1>
9753                              <1>     ; modified regs: esi, eax, edx, ch
9754                              <1>
9755 00003B75 EB87                <1>     jmp     short vbe_srs_retn
9756                              <1>
9757                              <1> ;vbe_srs_10:
9758                              <1> ;   ; successful
9759                              <1> ;   xor     eax, eax ; 0
9760                              <1> ;   mov     al, 4Fh ; ax = 004Fh (successful)
9761                              <1> ;   retn
9762                              <1>
9763                              <1> vbe_srs_gbs:
9764                              <1>     ; return buffer size according to flags
9765 00003B77 89CB                <1>     mov     ebx, ecx ; options/flags
9766 00003B79 D0E3                <1>     shl     bl, 1
9767 00003B7B 668B9B[833B0000]    <1>     mov     bx, [ebx+vbestatebufsize]
9768 00003B82 C3                  <1>     retn
9769                              <1>
9770                              <1> vbestatebufsize:
9771                              <1>     ; ---------------------------------------
9772                              <1>     ; CL = 0  1   2    3    4    5    6    7
9773                              <1>     ; ---------------------------------------
9774 00003B83 0000460028006E0004- <1>     dw      0, 70, 40, 110, 772, 842, 812, 882
9774 00003B8C 034A032C037203      <1>
9775                              <1>     ; ---------------------------------------
9776                              <1>     ; CL = 8   9   10  11   12   13   14   15
9777                              <1>     ; ---------------------------------------
9778 00003B93 120058003A00800016- <1>     dw      18, 88, 58, 128, 790, 860, 830, 900
9778 00003B9C 035C033E038403      <1>
9779                              <1>
9780                              <1> ; 11/01/2021
9781                              <1> VGAREG_ACTL_ADDRESS       equ 3C0h
9782                              <1> VGAREG_ACTL_WRITE_DATA    equ 3C0h
9783                              <1> VGAREG_ACTL_READ_DATA     equ 3C1h
9784                              <1>
9785                              <1> VGAREG_INPUT_STATUS       equ 3C2h
9786                              <1> VGAREG_WRITE_MISC_OUTPUT equ 3C2h
9787                              <1> VGAREG_VIDEO_ENABLE       equ 3C3h
9788                              <1> VGAREG_SEQU_ADDRESS       equ 3C4h
9789                              <1> VGAREG_SEQU_DATA   equ 3C5h
9790                              <1>
9791                              <1> VGAREG_PEL_MASK           equ 3C6h
9792                              <1> VGAREG_DAC_STATE   equ 3C7h
9793                              <1> VGAREG_DAC_READ_ADDRESS   equ 3C7h
9794                              <1> VGAREG_DAC_WRITE_ADDRESS equ 3C8h
9795                              <1> VGAREG_DAC_DATA           equ 3C9h
9796                              <1>
9797                              <1> VGAREG_READ_FEATURE_CTL   equ 3CAh
9798                              <1> VGAREG_READ_MISC_OUTPUT   equ 3CCh
9799                              <1>
9800                              <1> VGAREG_GRDC_ADDRESS       equ 3CEh
9801                              <1> VGAREG_GRDC_DATA   equ 3CFh
9802                              <1>
9803                              <1> ;VGAREG_MDA_CRTC_ADDRESS equ 3B4h
9804                              <1> ;VGAREG_MDA_CRTC_DATA     equ 3B5h
9805                              <1> VGAREG_VGA_CRTC_ADDRESS   equ 3D4h
9806                              <1> VGAREG_VGA_CRTC_DATA      equ 3D5h
9807                              <1>
9808                              <1> ;VGAREG_MDA_WRITE_FEATURE_CTL equ 3BAh
9809                              <1> VGAREG_VGA_WRITE_FEATURE_CTL equ 3DAh
9810                              <1> VGAREG_ACTL_RESET  equ 3DAh
9811                              <1>
9812                              <1> ;VGAREG_MDA_MODECTL       equ 3B8h
9813                              <1> VGAREG_CGA_MODECTL equ 3D8h
9814                              <1> VGAREG_CGA_PALETTE equ 3D9h
9815                              <1>
9816                              <1> biosfn_save_video_state:
9817                              <1>     ; 03/08/2022 (TRDOS 386 v2.0.5)
9818                              <1>     ; 22/01/2021
9819                              <1>     ; 12/01/2021
9820                              <1>     ; 11/01/2021 (TRDOS 386 v2.0.3)
9821                              <1>     ; (vgabios.c)
9822                              <1>
9823                              <1>     ; modified registers: eax, edx, edi, ch
9824                              <1>
9825                              <1>     ;mov    edi, VBE3SAVERESTOREBLOCK
9826                              <1>
9827                              <1>     ; input: edi = state buffer address
9828                              <1>
9829 00003BA3 F6C101              <1>     test    cl, 1
9830 00003BA6 0F8485000000        <1>     jz      bfn_svs_4
9831                              <1>
9832 00003BAC 66BAC403            <1>     mov     dx, VGAREG_SEQU_ADDRESS ; 3C7h
9833 00003BB0 EC                  <1>     in      al, dx
9834 00003BB1 AA                  <1>     stosb
9835                              <1>     ;mov    dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
9836 00003BB2 B2D4                <1>     mov     dl, 0D4h
9837 00003BB4 EC                  <1>     in      al, dx
9838 00003BB5 AA                  <1>     stosb
9839                              <1>     ;mov    dx, VGAREG_GRDC_ADDRESS ; 3CEh
9840 00003BB6 B2CE                <1>         mov dl, 0CEh
```

```
                        <1>     in      al, dx
                        <1>     stosb
                        <1>     ;mov    dx, VGAREG_ACTL_RESET ; 3DAh
                        <1>     mov     dl, 0DAh
                        <1>     in      al, dx
                        <1>     ;mov    dx, VGAREG_ACTL_ADDRESS ; 3C0h
                        <1>     mov     dl, 0C0h
                        <1>     in      al, dx
                        <1>     stosb
                        <1>     mov     ah, al ; ar_index
                        <1>     ;mov    dx, VGAREG_READ_FEATURE_CTL ; 3CAh
                        <1>     mov     dl, 0CAh
                        <1>     in      al, dx
                        <1>     stosb
                        <1>     ; (5 bytes are writen above)
                        <1>
                        <1>     ; for(i=1;i<=4;i++){
                        <1>     mov     al, 1
                        <1>     ;;mov   dx, VGAREG_SEQU_ADDRESS ; 3C4h
                        <1>     ;mov    dl, 0C4h
                        <1>     mov     ch, 4
                        <1> bfn_svs_0:
                        <1>     ; outb(VGAREG_SEQU_ADDRESS, i);
                        <1>     ;mov    dx, VGAREG_SEQU_ADDRESS ; 3C4h
                        <1>     mov     dl, 0C4h
                        <1>     out     dx, al
                        <1>     ;mov    dx, VGAREG_SEQU_DATA  ; 3C5h
                        <1>     inc     dl  ; dx = 3C5h
                        <1>     ; inb(VGAREG_SEQU_DATA)
                        <1>     push    eax
                        <1>     in      al, dx
                        <1>     stosb  ; (4 bytes in loop)
                        <1>     pop     eax
                        <1>     ;mov    dx, VGAREG_SEQU_ADDRESS ; 3C4h
                        <1>     ;dec    dl
                        <1>     inc     al  ; i++
                        <1>     dec     ch
                        <1>     jnz     short bfn_svs_0
                        <1>
                        <1>     ; outb(VGAREG_SEQU_ADDRESS, 0);
                        <1>     sub     al, al ; 0
                        <1>     out     dx, al
                        <1>     ; inb(VGAREG_SEQU_DATA)
                        <1>     ;mov    dx, VGAREG_SEQU_DATA ; 3C5h
                        <1>     inc     dl  ; dx = 3C5h
                        <1>     in      al, dx
                        <1>     stosb  ; (+1 byte)
                        <1>
                        <1>         ; for(i=0;i<=0x18;i++) {
                        <1>     sub     al, al ; 0
                        <1>     ;;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
                        <1>     ;mov    dl, 0D4h
                        <1>     mov     ch, 25
                        <1> bfn_svs_1:
                        <1>         ; outb(crtc_addr,i);
                        <1>     ;mov    dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
                        <1>     mov     dl, 0D4h
                        <1>     out     dx, al
                        <1>     ;mov    dx, VGAREG_VGA_CRTC_DATA ; 3D5h
                        <1>     inc     dl  ; dx = 3D5h
                        <1>     ; inb(crtc_addr+1)
                        <1>     push    eax
                        <1>     in      al, dx
                        <1>     stosb  ; (25 bytes in loop)
                        <1>     pop     eax
                        <1>     ;mov    dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
                        <1>     ;dec    dl
                        <1>     inc     al  ; i++
                        <1>     dec     ch
                        <1>     jnz     short bfn_svs_1
                        <1>
                        <1>     and     ah, 20h  ; (ar_index & 0x20)
                        <1>         ; for(i=0;i<=0x13;i++) {
                        <1>     sub     al, al ; 0
                        <1>     mov     ch, 20
                        <1> bfn_svs_2:
                        <1>     ; inb(VGAREG_ACTL_RESET);
                        <1>     ;mov    dx, VGAREG_ACTL_RESET ; 3DAh
                        <1>     mov     dl, 0DAh
                        <1>     push    eax
                        <1>     in      al, dx
                        <1>     mov     al, [esp]
                        <1>     ; outb(VGAREG_ACTL_ADDRESS, i | (ar_index & 0x20));
                        <1>     or      al, ah
                        <1>     ;mov    dx, VGAREG_ACTL_ADDRESS ; 3C0h
                        <1>     mov     dl, 0C0h
                        <1>     out     dx, al
                        <1>     ;mov    dx, VGAREG_ACTL_READ_DATA ; 3C1h
                        <1>     ;mov    dl, 0C1h
                        <1>     inc     dl
                        <1>     in      al, dx
                        <1>     stosb  ; (20 bytes in loop)
                        <1>     pop     eax
                        <1>     inc     al  ; i++
                        <1>     dec     ch
                        <1>     jnz     short bfn_svs_2
                        <1>
                        <1>     ; inb(VGAREG_ACTL_RESET);
                        <1>     ;mov    dx, VGAREG_ACTL_RESET ; 3DAh
                        <1>     mov     dl, 0DAh
                        <1>     in      al, dx
                        <1>
                        <1>         ; for(i=0;i<=8;i++) {
                        <1>     sub     al, al ; 0
                        <1>     ;;mov   dx, VGAREG_GRDC_ADDRESS ; 3CEh
                        <1>     ;mov    dl, 0CEh
                        <1>     mov     ch, 9
                        <1> bfn_svs_3:
                        <1>     ; outb(VGAREG_GRDC_ADDRESS,i)
                        <1>     ;mov    dx, VGAREG_GRDC_ADDRESS ; 3CEh
                        <1>     mov     dl, 0CEh
                        <1>     out     dx, al
                        <1>     ; inb(VGAREG_ACTL_READ_DATA)
                        <1>     push    eax
                        <1>     ;mov    dx, VGAREG_GRDC_DATA ; 3CFh
                        <1>     ;mov    dl, 0CFh
                        <1>     inc     dl
                        <1>     in      al, dx
                        <1>     stosb  ; (9 bytes in loop)
                        <1>     pop     eax
                        <1>     ;dec    dl
                        <1>     inc     al  ; i++
                        <1>     dec     ch
                        <1>     jnz     short bfn_svs_3
```

```
                                <1>
                                <1>     ; write_word(ES, BX, crtc_addr); BX+= 2;
                                <1>     ; (offset 64)
00003C28 66B8D403               <1>     mov     ax, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
00003C2C 66AB                   <1>     stosw   ; (2 bytes (1 word))
                                <1>
                                <1>         ; /* XXX: read plane latches */
00003C2E 31C0                   <1>     xor     eax, eax  ; 0
00003C30 AB                     <1>             stosd   ; (4 bytes)
                                <1>
                                <1>     ; (total 70 bytes are written above as controller hardware state)
                                <1>
                                <1> bfn_svs_4:
                                <1>     ; 12/01/2021 (TRDOS 386 v2.0.3)
00003C31 F6C102                 <1>     test    cl, 2
00003C34 7476                   <1>     jz      short bfn_svs_6
                                <1>
                                <1>     ; VIDEO BIOS DATA
                                <1>     ; !!! this data is valid for TRDOS 386 v2 kernel only !!!
                                <1>     ; (this is not same with BOCHS/PLEX86 video bios, BIOS data)
                                <1>
                                <1>     ; if (CX & 2) {
                                <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_CURRENT_MODE)); BX++;
                                <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS)); BX += 2;
                                <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_PAGE_SIZE)); BX += 2;
                                <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_CRTC_ADDRESS)); BX += 2;
                                <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS)); BX++;
                                <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_CHAR_HEIGHT)); BX += 2;
                                <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_VIDEO_CTL)); BX++;
                                <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_SWITCHES)); BX++;
                                <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_MODESET_CTL)); BX++;
                                <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_CURSOR_TYPE)); BX += 2;
                                <1>         ;for(i=0;i<8;i++) {
                                <1>         ;   write_word(ES, BX, read_word(BIOSMEM_SEG, BIOSMEM_CURSOR_POS+2*i));
                                <1>         ;   BX += 2;
                                <1>         ;}
                                <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_CURRENT_START)); BX += 2;
                                <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_CURRENT_PAGE)); BX++;
                                <1>         ;/* current font */
                                <1>         ;write_word(ES, BX, read_word(0, 0x1f * 4)); BX += 2;
                                <1>         ;write_word(ES, BX, read_word(0, 0x1f * 4 + 2)); BX += 2;
                                <1>         ;write_word(ES, BX, read_word(0, 0x43 * 4)); BX += 2;
                                <1>         ;write_word(ES, BX, read_word(0, 0x43 * 4 + 2)); BX += 2;
                                <1>
                                <1>     ; !!! save TRDOS 386 v2 kernel spesific video bios data !!!
                                <1>     ; (which is/are used by 'SET_MODE' function and/or it's sub functions)
                                <1>
00003C36 66B8D403               <1>     mov     ax, 3D4h ; CRTC_ADDR, always 3D4h (color VGA) for TRDOS 386 v2
00003C3A 66AB                   <1>     stosw
00003C3C A0[1E680000]           <1>     mov     al, [CRT_MODE] ; Current video mode (0FFh for VESA VBE modes)
00003C41 AA                     <1>     stosb
00003C42 A0[1F680000]           <1>     mov     al, [CRT_MODE_SET] ; 29h for mode 03h ; TRDOS 386 feature only !
00003C47 AA                     <1>     stosb
00003C48 66A1[EE9C0100]         <1>     mov     ax, [video_mode] ; Current VESA VBE (SVGA, extended VGA) mode
00003C4E 66AB                   <1>     stosw                   ; (valid if [CRT_MODE] = 0FFh)
00003C50 66A1[18830100]         <1>     mov     ax, [CRT_LEN] ; page size (in bytes)
00003C56 66AB                   <1>     stosw
00003C58 66A1[9C760100]         <1>     mov     ax, [CRT_START] ; video page start offset
00003C5E 66AB                   <1>     stosw
00003C60 A0[20680000]           <1>     mov     al, [CRT_COLS] ; nbcols, characters per row
00003C65 AA                     <1>     stosb
00003C66 A0[26680000]           <1>     mov     al, [VGA_ROWS] ; nbrows, (character) rows per page (not rows-1)
00003C6B AA                     <1>     stosb
00003C6C A0[22680000]           <1>     mov     al, [CHAR_HEIGHT] ; character font height (8 or 16 or 14)
00003C71 AA                     <1>     stosb
00003C72 A0[23680000]           <1>     mov     al, [VGA_VIDEO_CTL] ; ROM BIOS DATA AREA Offset 87h
00003C77 AA                     <1>     stosb
00003C78 A0[24680000]           <1>     mov     al, [VGA_SWITCHES] ; feature bit switches
00003C7D AA                     <1>     stosb
00003C7E A0[25680000]           <1>     mov     al, [VGA_MODESET_CTL] ; basic mode set options
00003C83 AA                     <1>     stosb
                                <1>     ; followings are only used by TRDOS 386 v2 (IBM PC/AT ROMBIOS) code
                                <1>     ; (bochs/plex86 does not use and return those)
00003C84 A0[21680000]           <1>     mov     al, [CRT_PALETTE] ; current color palette ; TRDOS 386 feature only !
00003C89 AA                     <1>     stosb
00003C8A A0[AE760100]           <1>     mov     al, [ACTIVE_PAGE] ; current video page
00003C8F AA                     <1>     stosb
00003C90 66A1[37680000]         <1>     mov     ax, [CURSOR_MODE] ; cursor type
00003C96 66AB                   <1>     stosw
                                <1>     ;mov    eax, [CURSOR_POSN] ; cursor position for video page 0 and 1
                                <1>     ;stosd
                                <1>     ;mov    eax, [CURSOR_POSN+4] ; cursor position for video page 2 and 3
                                <1>     ;stosd
                                <1>     ;mov    eax, [CURSOR_POSN+8] ; cursor position for video page 4 and 5
                                <1>     ;stosd
                                <1>     ;mov    eax, [CURSOR_POSN+12] ; cursor position for video page 6 and 7
                                <1>     ;stosd
00003C98 56                     <1>     push    esi
00003C99 B504                   <1>     mov     ch, 4
00003C9B BE[9E760100]           <1>     mov     esi, CURSOR_POSN
                                <1> bfn_svs_5:
00003CA0 A5                     <1>     movsd
00003CA1 FECD                   <1>     dec     ch
00003CA3 75FB                   <1>     jnz     short bfn_svs_5
00003CA5 5E                     <1>     pop     esi
                                <1>     ; (font addr) protected mode address in kernel's/system memory space
                                <1>     ; (not accessable/meaningful address value by user)
00003CA6 A1[2A830100]           <1>     mov     eax, [VGA_INT43H] ; VGA current (default) font address
00003CAB AB                     <1>     stosd
                                <1>
                                <1>     ; (total 40 bytes are written above as BIOS data state)
                                <1>
                                <1> bfn_svs_6:
                                <1>     ; 12/01/2021
00003CAC F6C104                 <1>     test    cl, 4
00003CAF 7422                   <1>     jz      short bfn_svs_8
                                <1>
                                <1>     ;/* XXX: check this */
                                <1>             ; /* read/write mode dac */
                                <1>         ;write_byte(ES, BX, inb(VGAREG_DAC_STATE)); BX++;
                                <1>     ;       ; /* pix address */
                                <1>         ;write_byte(ES, BX, inb(VGAREG_DAC_WRITE_ADDRESS)); BX++;
                                <1>         ;write_byte(ES, BX, inb(VGAREG_PEL_MASK)); BX++;
                                <1>         ;// Set the whole dac always, from 0
                                <1>         ;outb(VGAREG_DAC_WRITE_ADDRESS,0x00);
                                <1>         ;for(i=0;i<256*3;i++) {
                                <1>         ;   write_byte(ES, BX, inb(VGAREG_DAC_DATA)); BX++;
                                <1>         ;}
                                <1>         ;write_byte(ES, BX, 0); BX++; /* color select register */
                                <1>
                                <1>     ; /* read/write mode dac */
00003CB1 66BAC703               <1>     mov     dx, 3C7h ; VGAREG_DAC_STATE
00003CB5 EC                     <1>     in      al, dx
00003CB6 AA                     <1>     stosb
```

```
10089                                   <1>     ; /* pix address */
10090                                   <1>     ;mov    dx, VGAREG_DAC_WRITE_ADDRESS ; 3C8h
10091                                   <1>     ;mov    dl, 0C8h
10092 00003CB7 FEC2                     <1>     inc     dl
10093 00003CB9 EC                       <1>     in      al, dx
10094 00003CBA AA                       <1>     stosb
10095                                   <1>     ;mov    dx, VGAREG_PEL_MASK  ; 3C6h
10096 00003CBB B2C6                     <1>     mov     dl, 0C6h
10097 00003CBD EC                       <1>     in      al, dx
10098 00003CBE AA                       <1>     stosb
10099                                   <1>     ;// Set the whole dac always, from 0
10100 00003CBF 30C0                     <1>     xor     al, al ; 0
10101                                   <1>     ;mov    dx, VGAREG_DAC_WRITE_ADDRESS ; 3C8h
10102 00003CC1 B2C8                     <1>     mov     dl, 0C8h
10103 00003CC3 EE                       <1>     out     dx, al
10104                                   <1>
10105 00003CC4 51                       <1>     push    ecx ; 22/01/2021
10106                                   <1>     ;for(i=0;i<256*3;i++) {
10107                                   <1>     ;mov    ecx, 256*3 ; 768 bytes
10108                                   <1>     ; 03/08/2022
10109 00003CC5 29C9                     <1>     sub     ecx, ecx
10110 00003CC7 B503                     <1>     mov     ch, 3
10111                                   <1>     ; ecx = 300h = 768
10112                                   <1>     ;mov    dx, VGAREG_DAC_DATA ; 3C9h
10113                                   <1>     ;mov    dl, 0C9h
10114 00003CC9 FEC2                     <1>     inc     dl ; dx = 3C9h
10115                                   <1> bfn_svs_7:
10116 00003CCB EC                       <1>     in      al, dx
10117 00003CCC AA                       <1>     stosb
10118 00003CCD E2FC                     <1>     loop    bfn_svs_7
10119 00003CCF 59                       <1>     pop     ecx ; 22/01/2021
10120                                   <1>
10121                                   <1>     ; /* color select register */
10122 00003CD0 28C0                     <1>     sub     al, al ; 0
10123 00003CD2 AA                       <1>     stosb
10124                                   <1>
10125                                   <1>     ; (total 772 bytes are written above as DAC state)
10126                                   <1> bfn_svs_8:
10127 00003CD3 C3                       <1>     retn
10128                                   <1>
10129                                   <1> vbe_biosfn_save_video_state:
10130                                   <1>     ; 23/01/2021
10131                                   <1>     ; 13/01/2021
10132                                   <1>     ; 12/01/2021 (TRDOS 386 v2.0.3)
10133                                   <1>     ; (vbe.c)
10134                                   <1>
10135                                   <1>     ; modified registers: eax, edx, edi, ch
10136                                   <1>
10137                                   <1>     ; input: edi = state buffer address
10138                                   <1>     ; output:
10139                                   <1>     ;       VBE DISPI register contents will be saved
10140                                   <1>     ;       (18 bytes, 9 words)
10141                                   <1>
10142                                   <1>     ; outw(VBE_DISPI_IOPORT_INDEX,VBE_DISPI_INDEX_ENABLE);
10143                                   <1>     ; enable = inw(VBE_DISPI_IOPORT_DATA);
10144                                   <1>     ; write_word(ES, BX, enable);
10145                                   <1>     ; BX += 2;
10146                                   <1>     ; if (!(enable & VBE_DISPI_ENABLED))
10147                                   <1>     ;       return;
10148                                   <1>     ; for(i = VBE_DISPI_INDEX_XRES;
10149                                   <1>     ;                 i <= VBE_DISPI_INDEX_Y_OFFSET; i++) {
10150                                   <1>     ;    if (i != VBE_DISPI_INDEX_ENABLE) {
10151                                   <1>     ;        outw(VBE_DISPI_IOPORT_INDEX, i);
10152                                   <1>     ;        write_word(ES, BX, inw(VBE_DISPI_IOPORT_DATA));
10153                                   <1>     ;    BX += 2;
10154                                   <1>     ;        }
10155                                   <1>     ; }
10156                                   <1>
10157 00003CD4 66BACE01                 <1>     mov     dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
10158                                   <1>     ;mov    eax, 04h  ; VBE_DISPI_INDEX_ENABLE
10159                                   <1>      ;03/08/2022
10160 00003CD8 66EF                     <1>     out     dx, ax
10161                                   <1>     ;mov    dx, 01CFh ; VBE_DISPI_IOPORT_DATA
10162 00003CDA FEC2                     <1>     inc     dl
10163 00003CDC 66ED                     <1>     in      ax, dx ; enable (status)
10164 00003CDE 66AB                     <1>     stosw
10165 00003CE0 6683E001                 <1>     and     ax, 1 ; VBE_DISPI_ENABLED
10166 00003CE4 7505                     <1>     jnz     short vbe_bfn_svs_0
10167                                   <1>     ; 23/01/2021
10168                                   <1>     ; ax = 0
10169                                   <1>     ; VBE_DISPI_DISABLED
10170                                   <1>     ; 13/01/2021
10171                                   <1>     ; clear remain 8 bytes
10172                                   <1>     ;xor    eax, eax
10173 00003CE6 AB                       <1>     stosd   ; 2
10174 00003CE7 AB                       <1>     stosd   ; 2
10175 00003CE8 AB                       <1>     stosd   ; 2
10176 00003CE9 AB                       <1>     stosd   ; 2
10177 00003CEA C3                       <1>     retn
10178                                   <1> vbe_bfn_svs_0:
10179                                   <1>     ; VBE_DISPI_ENABLED
10180                                   <1>
10181                                   <1>     ;sub    eax, eax
10182 00003CEB 28C0                     <1>     sub     al, al ; eax = 0
10183                                   <1>
10184                                   <1>     ; from VBE_DISPI_INDEX_XRES
10185                                   <1>     ;   to VBE_DISPI_INDEX_BPP
10186                                   <1>
10187 00003CED B503                     <1>     mov     ch, 3
10188                                   <1>     ; al = 0 ; VBE_DISPI_INDEX_XRES - 1
10189                                   <1>
10190 00003CEF E804000000               <1>     call    vbe_bfn_svs_1
10191                                   <1>
10192                                   <1>     ; from VBE_DISPI_INDEX_BANK
10193                                   <1>     ;   to VBE_DISPI_INDEX_Y_OFFSET
10194                                   <1>
10195 00003CF4 FEC0                     <1>     inc     al
10196                                   <1>     ; al = 4 ; VBE_DISPI_INDEX_BANK - 1
10197                                   <1>
10198 00003CF6 B505                     <1>     mov     ch, 5
10199                                   <1> vbe_bfn_svs_1:
10200 00003CF8 FEC0                     <1>     inc     al ; from VBE_DISPI_INDEX_XRES
10201                                   <1>                ;   to VBE_DISPI_INDEX_BPP
10202                                   <1>     ;mov    dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
10203 00003CFA FECA                     <1>     dec     dl ; 1CEh
10204 00003CFC 66EF                     <1>     out     dx, ax
10205 00003CFE 50                       <1>     push    eax
10206                                   <1>     ;mov    dx, 01CFh ; VBE_DISPI_IOPORT_DATA
10207 00003CFF FEC2                     <1>     inc     dl ; 1CFh
10208 00003D01 66ED                     <1>     in      ax, dx
10209 00003D03 66AB                     <1>     stosw
10210 00003D05 58                       <1>     pop     eax
10211 00003D06 FECD                     <1>     dec     ch
10212 00003D08 75EE                     <1>     jnz     short vbe_bfn_svs_1
```

```
10213 00003D0A C3                <1>     retn
10214                            <1>
10215                            <1> biosfn_restore_video_state:
10216                            <1>     ; 22/01/2021
10217                            <1>     ; 13/01/2021
10218                            <1>     ; 12/01/2021 (TRDOS 386 v2.0.3)
10219                            <1>     ; (vgabios.c)
10220                            <1>
10221                            <1>     ; modified registers: eax, edx, esi, edi, ch
10222                            <1>
10223                            <1>     ;mov    esi, VBE3SAVERESTOREBLOCK
10224                            <1>
10225                            <1>     ; input: esi = state buffer address
10226                            <1>
10227 00003D0B F6C101            <1>     test    cl, 1
10228 00003D0E 0F84A9000000      <1>     jz      bfn_rvs_6
10229                            <1>
10230 00003D14 66817E40D403      <1>     cmp     word [esi+64], 3D4h ; must be 3D4h
10231 00003D1A 7402              <1>     je      short bfn_rvs_0
10232                            <1>                     ; it is seen as valid buffer
10233 00003D1C F9                <1>     stc
10234 00003D1D C3                <1>     retn
10235                            <1>
10236                            <1> bfn_rvs_0:
10237 00003D1E 89F7              <1>     mov     edi, esi ; addr1
10238 00003D20 83C605            <1>     add     esi, 5 ; skip 1st 5 bytes for now
10239                            <1>
10240                            <1>     ; // Reset Attribute Ctl flip-flop
10241                            <1>         ; inb(VGAREG_ACTL_RESET);
10242 00003D23 66BADA03          <1>     mov     dx, 3DAh ; VGAREG_ACTL_RESET
10243 00003D27 EC                <1>     in      al, dx
10244                            <1>
10245                            <1>     ; for(i=1;i<=4;i++){
10246 00003D28 B001              <1>     mov     al, 1
10247                            <1>     ;;mov   dx, VGAREG_SEQU_ADDRESS ; 3C4h
10248                            <1>     ;mov    dl, 0C4h
10249 00003D2A B504              <1>     mov     ch, 4
10250                            <1> bfn_rvs_1:
10251                            <1>     ; outb(VGAREG_SEQU_ADDRESS, i);
10252                            <1>     ;mov    dx, VGAREG_SEQU_ADDRESS ; 3C4h
10253 00003D2C B2C4              <1>     mov     dl, 0C4h
10254 00003D2E EE                <1>     out     dx, al
10255                            <1>     ;mov    dx, VGAREG_SEQU_DATA  ; 3C5h
10256 00003D2F FEC2              <1>     inc     dl  ; dx = 3C5h
10257                            <1>     ; outb(VGAREG_SEQU_DATA)
10258 00003D31 50                <1>     push    eax
10259 00003D32 AC                <1>     lodsb  ; (4 bytes in loop)
10260 00003D33 EE                <1>     out     dx, al
10261 00003D34 58                <1>     pop     eax
10262                            <1>     ;mov    dx, VGAREG_SEQU_ADDRESS ; 3C4h
10263                            <1>     ;dec    dl
10264 00003D35 FEC0              <1>     inc     al  ; i++
10265 00003D37 FECD              <1>     dec     ch
10266 00003D39 75F1              <1>     jnz     short bfn_rvs_1
10267                            <1>
10268                            <1>     ; outb(VGAREG_SEQU_ADDRESS, 0);
10269 00003D3B 28C0              <1>     sub     al, al ; 0
10270 00003D3D EE                <1>     out     dx, al
10271                            <1>     ; outb(VGAREG_SEQU_DATA)
10272                            <1>     ;mov    dx, VGAREG_SEQU_DATA ; 3C5h
10273 00003D3E FEC2              <1>     inc     dl  ; dx = 3C5h
10274 00003D40 AC                <1>     lodsb  ; (+1 byte)
10275 00003D41 EE                <1>     out     dx, al
10276                            <1>
10277                            <1>     ; // Disable CRTC write protection
10278                            <1>     ; outw(crtc_addr,0x0011);
10279                            <1>     ;mov    dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
10280 00003D42 B2D4              <1>     mov     dl, 0D4h
10281 00003D44 66B81100          <1>     mov     ax, 11h
10282 00003D48 66EF              <1>     out     dx, ax
10283                            <1>
10284                            <1>     ; // Set CRTC regs
10285                            <1>
10286                            <1>     ; for(i=0;i<=0x18;i++) {
10287                            <1>         ;   if (i != 0x11) {
10288 00003D4A 28C0              <1>     sub     al, al ; 0
10289                            <1>     ;;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
10290                            <1>     ;mov    dl, 0D4h
10291 00003D4C B519              <1>     mov     ch, 25
10292                            <1> bfn_rvs_2:
10293                            <1>         ; outb(crtc_addr,i);
10294                            <1>     ;mov    dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
10295 00003D4E B2D4              <1>     mov     dl, 0D4h
10296 00003D50 EE                <1>     out     dx, al
10297                            <1>     ;mov    dx, VGAREG_VGA_CRTC_DATA ; 3D5h
10298 00003D51 FEC2              <1>     inc     dl  ; dx = 3D5h
10299                            <1>     ; inb(crtc_addr+1)
10300 00003D53 50                <1>     push    eax
10301 00003D54 AC                <1>     lodsb  ; (25 bytes in loop)
10302 00003D55 EE                <1>     out     dx, al
10303 00003D56 58                <1>     pop     eax
10304                            <1>     ;mov    dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
10305                            <1>     ;dec    dl
10306 00003D57 FEC0              <1>     inc     al  ; i++
10307 00003D59 3C11              <1>     cmp     al, 17 ; 11h
10308 00003D5B 7505              <1>     jne     short bfn_rvs_3
10309 00003D5D AC                <1>     lodsb
10310 00003D5E 88C4              <1>     mov     ah, al ; *
10311 00003D60 B012              <1>     mov     al, 18
10312                            <1> bfn_rvs_3:
10313 00003D62 FECD              <1>     dec     ch
10314 00003D64 75E8              <1>     jnz     short bfn_rvs_2
10315                            <1>
10316                            <1>     ; // select crtc base address
10317                            <1>         ; v = inb(VGAREG_READ_MISC_OUTPUT) & ~0x01;
10318                            <1>     ;if (crtc_addr = 0x3d4)
10319                            <1>     ;   v |= 0x01;
10320                            <1>     ; outb(VGAREG_WRITE_MISC_OUTPUT, v);
10321                            <1>
10322                            <1>     ;;mov   dx, VGAREG_READ_MISC_OUTPUT ; 3CCh
10323                            <1>     ;mov    dl, 0CCh
10324                            <1>     ;in     al, dl
10325                            <1>     ;and    al, 1
10326                            <1>     ;;mov   dx, VGAREG_WRITE_MISC_OUTPUT ; 3C2h
10327                            <1>     ;mov    dl, 0C2h
10328                            <1>     ;or     al, 1
10329                            <1>     ;out    dx, al
10330                            <1>
10331                            <1>     ; // enable write protection if needed
10332                            <1>     ;outb(crtc_addr, 0x11);
10333                            <1>         ;outb(crtc_addr+1, read_byte(ES, BX - 0x18 + 0x11));
10334                            <1>     ;mov    dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
10335 00003D66 B2D4              <1>     mov     dl, 0D4h
10336 00003D68 B011              <1>     mov     al, 11h
```

```
10337 00003D6A EE                <1>     out     dx, al
10338 00003D6B 88E0              <1>     mov     al, ah ; *
10339 00003D6D FEC2              <1>     inc     dl ; dx = 3D5h
10340 00003D6F EE                <1>     out     dx, al
10341                            <1>
10342                            <1>     ; // Set Attribute Ctl
10343 00003D70 8A6703            <1>     mov     ah, [edi+3] ; addr1+3, ah = ar_index
10344 00003D73 80E420            <1>     and     ah, 20h  ; (ar_index & 0x20)
10345                            <1>
10346                            <1>         ; inb(VGAREG_ACTL_RESET);
10347                            <1>     ;mov    dx, 3DAh ; VGAREG_ACTL_RESET
10348 00003D76 B2DA              <1>     mov     dl, 0DAh
10349 00003D78 EC                <1>     in      al, dx
10350                            <1>
10351                            <1>         ; for(i=0;i<=0x13;i++) {
10352 00003D79 28C0              <1>     sub     al, al ; 0
10353 00003D7B B514              <1>     mov     ch, 20
10354                            <1> bfn_rvs_4:
10355                            <1>     ; outb(VGAREG_ACTL_ADDRESS, i | (ar_index & 0x20));
10356 00003D7D 50                <1>     push    eax
10357 00003D7E 08E0              <1>     or      al, ah
10358                            <1>     ;mov    dx, VGAREG_ACTL_ADDRESS ; 3C0h
10359 00003D80 B2C0              <1>     mov     dl, 0C0h
10360 00003D82 EE                <1>     out     dx, al
10361                            <1>     ;mov    dx, VGAREG_ACTL_WRITE_DATA ; 3C0h
10362                            <1>     ;mov    dl, 0C0h
10363 00003D83 AC                <1>     lodsb  ; (20 bytes in loop)
10364 00003D84 EE                <1>     out     dx, al
10365 00003D85 58                <1>     pop     eax
10366 00003D86 FEC0              <1>     inc     al  ; i++
10367 00003D88 FECD              <1>     dec     ch
10368 00003D8A 75F1              <1>     jnz     short bfn_rvs_4
10369                            <1>
10370                            <1>     ; outb(VGAREG_ACTL_ADDRESS, ar_index);
10371                            <1>     ;mov    dx, VGAREG_ACTL_ADDRESS ; 3C0h
10372                            <1>     ;mov    dl, 0C0h
10373 00003D8C 88E0              <1>     mov     al, ah ; ar_index
10374 00003D8E EE                <1>     out     dx, al
10375                            <1>
10376                            <1>     ; inb(VGAREG_ACTL_RESET);
10377                            <1>     ;mov    dx, VGAREG_ACTL_RESET ; 3DAh
10378 00003D8F B2DA              <1>     mov     dl, 0DAh
10379 00003D91 EC                <1>     in      al, dx
10380                            <1>
10381                            <1>         ; for(i=0;i<=8;i++) {
10382 00003D92 28C0              <1>     sub     al, al ; 0
10383                            <1>     ;;mov   dx, VGAREG_GRDC_ADDRESS ; 3CEh
10384                            <1>     ;mov    dl, 0CEh
10385 00003D94 B509              <1>     mov     ch, 9
10386                            <1> bfn_rvs_5:
10387                            <1>     ; outb(VGAREG_GRDC_ADDRESS,i)
10388                            <1>     ;mov    dx, VGAREG_GRDC_ADDRESS ; 3CEh
10389 00003D96 B2CE              <1>     mov     dl, 0CEh
10390 00003D98 EE                <1>     out     dx, al
10391                            <1>     ; outb(VGAREG_ACTL_READ_DATA)
10392 00003D99 50                <1>     push    eax
10393                            <1>     ;mov    dx, VGAREG_GRDC_DATA ; 3CFh
10394                            <1>     ;mov    dl, 0CFh
10395 00003D9A FEC2              <1>     inc     dl
10396 00003D9C AC                <1>     lodsb  ; (9 bytes in loop)
10397 00003D9D EE                <1>     out     dx, al
10398 00003D9E 58                <1>     pop     eax
10399                            <1>     ;dec    dl
10400 00003D9F FEC0              <1>     inc     al  ; i++
10401 00003DA1 FECD              <1>     dec     ch
10402 00003DA3 75F1              <1>     jnz     short bfn_rvs_5
10403                            <1>
10404                            <1>     ; BX += 2; /* crtc_addr */     ; 3D4h
10405                            <1>         ; BX += 4; /* plane latches */ ; 0
10406 00003DA5 83C606            <1>     add     esi, 6
10407 00003DA8 56                <1>     push    esi ; *
10408                            <1>
10409                            <1>     ;outb(VGAREG_SEQU_ADDRESS, read_byte(ES, addr1)); addr1++;
10410                            <1>         ;outb(crtc_addr, read_byte(ES, addr1)); addr1++;
10411                            <1>         ;outb(VGAREG_GRDC_ADDRESS, read_byte(ES, addr1)); addr1++;
10412                            <1>         ;addr1++;
10413                            <1>         ;outb(crtc_addr - 0x4 + 0xa, read_byte(ES, addr1)); addr1++;
10414                            <1>
10415 00003DA9 89FE              <1>     mov     esi, edi ; start of state buffer
10416                            <1>
10417                            <1>     ;mov    dx, VGAREG_SEQU_ADDRESS ; 3C7h
10418 00003DAB B2C7              <1>     mov     dl, 0C7h
10419 00003DAD AC                <1>     lodsb
10420 00003DAE EE                <1>     out     dx, al
10421                            <1>     ;mov    dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
10422 00003DAF B2D4              <1>     mov     dl, 0D4h
10423 00003DB1 AC                <1>     lodsb
10424 00003DB2 EE                <1>     out     dx, al
10425                            <1>     ;mov    dx, VGAREG_GRDC_ADDRESS ; 3CEh
10426 00003DB3 B2CE              <1>         mov dl, 0CEh
10427 00003DB5 AC                <1>     lodsb
10428 00003DB6 EE                <1>     out     dx, al
10429 00003DB7 AC                <1>     lodsb  ; addr1++
10430                            <1>     ;mov    dx, VGAREG_VGA_WRITE_FEATURE_CTL ; 3DAh
10431 00003DB8 B2DA              <1>     mov     dl, 0DAh
10432 00003DBA AC                <1>     lodsb
10433 00003DBB EE                <1>     out     dx, al
10434                            <1>
10435 00003DBC 5E                <1>     pop     esi ; *
10436                            <1>
10437                            <1>     ; (total 70 bytes are read above as controller hardware state)
10438                            <1>
10439                            <1> bfn_rvs_6:
10440                            <1>     ; 13/01/2021
10441 00003DBD F6C102            <1>     test    cl, 2
10442 00003DC0 747D              <1>     jz      short bfn_rvs_9
10443                            <1>
10444                            <1>     ; VIDEO BIOS DATA
10445                            <1>     ; !!! this data is valid for TRDOS 386 v2 kernel only !!!
10446                            <1>     ; (this is not same with BOCHS/PLEX86 video bios, BIOS data)
10447                            <1>
10448                            <1>     ; if (CX & 2) {
10449                            <1>     ;write_byte(BIOSMEM_SEG,BIOSMEM_CURRENT_MODE, read_byte(ES, BX)); BX++;
10450                            <1>         ;write_word(BIOSMEM_SEG,BIOSMEM_NB_COLS, read_word(ES, BX)); BX += 2;
10451                            <1>         ;write_word(BIOSMEM_SEG,BIOSMEM_PAGE_SIZE, read_word(ES, BX)); BX += 2;
10452                            <1>         ;write_word(BIOSMEM_SEG,BIOSMEM_CRTC_ADDRESS, read_word(ES, BX)); BX += 2;
10453                            <1>         ;write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, read_byte(ES, BX)); BX++;
10454                            <1>         ;write_word(BIOSMEM_SEG,BIOSMEM_CHAR_HEIGHT, read_word(ES, BX)); BX += 2;
10455                            <1>         ;write_byte(BIOSMEM_SEG,BIOSMEM_VIDEO_CTL, read_byte(ES, BX)); BX++;
10456                            <1>         ;write_byte(BIOSMEM_SEG,BIOSMEM_SWITCHES, read_byte(ES, BX)); BX++;
10457                            <1>         ;write_byte(BIOSMEM_SEG,BIOSMEM_MODESET_CTL, read_byte(ES, BX)); BX++;
10458                            <1>         ;write_word(BIOSMEM_SEG,BIOSMEM_CURSOR_TYPE, read_word(ES, BX)); BX += 2;
10459                            <1>         ;for(i=0;i<8;i++) {
10460                            <1>         ;   write_word(BIOSMEM_SEG, BIOSMEM_CURSOR_POS+2*i, read_word(ES, BX));
```

```
10461                                  <1>         ;   BX += 2;
10462                                  <1>         ;}
10463                                  <1>         ;write_word(BIOSMEM_SEG,BIOSMEM_CURRENT_START, read_word(ES, BX)); BX += 2;
10464                                  <1>         ;write_byte(BIOSMEM_SEG,BIOSMEM_CURRENT_PAGE, read_byte(ES, BX)); BX++;
10465                                  <1>         ;/* current font */
10466                                  <1>         ;write_word(0, 0x1f * 4, read_word(ES, BX)); BX += 2;
10467                                  <1>         ;write_word(0, 0x1f * 4 + 2, read_word(ES, BX)); BX += 2;
10468                                  <1>         ;write_word(0, 0x43 * 4, read_word(ES, BX)); BX += 2;
10469                                  <1>         ;write_word(0, 0x43 * 4 + 2, read_word(ES, BX)); BX += 2;
10470                                  <1>
10471                                  <1>     ; !!! save TRDOS 386 v2 kernel spesific video bios data !!!
10472                                  <1>     ; (which is/are used by 'SET_MODE' function and/or it's sub functions)
10473                                  <1>
10474 00003DC2 66AD                   <1>     lodsw   ; CRTC_ADDR, always 3D4h (color VGA) for TRDOS 386 v2
10475                                  <1>     ; skip 3D4h check if it is already checked
10476 00003DC4 F6C101                 <1>     test    cl, 1
10477 00003DC7 7508                   <1>     jnz     short bfn_rvs_7
10478 00003DC9 663DD403               <1>     cmp     ax, 3D4h
10479 00003DCD 7402                   <1>     je      short bfn_rvs_7
10480 00003DCF F9                     <1>     stc
10481 00003DD0 C3                     <1>     retn
10482                                  <1> bfn_rvs_7:
10483 00003DD1 AC                     <1>     lodsb
10484 00003DD2 A2[1E680000]           <1>     mov     [CRT_MODE], al ; Current video mode (0FFh for VESA VBE modes)
10485 00003DD7 AC                     <1>     lodsb
10486 00003DD8 A2[1F680000]           <1>     mov     [CRT_MODE_SET], al ; 29h for mode 03h ; TRDOS 386 feature only !
10487 00003DDD 66AD                   <1>     lodsw
10488 00003DDF 66A3[EE9C0100]         <1>     mov     [video_mode], ax ; Current VESA VBE (SVGA, extended VGA) mode
10489 00003DE5 66AD                   <1>     lodsw           ; (valid if [CRT_MODE] = 0FFh)
10490 00003DE7 66A3[18830100]         <1>     mov     [CRT_LEN], ax ; page size (in bytes)
10491 00003DED 66AD                   <1>     lodsw
10492 00003DEF 66A3[9C760100]         <1>     mov     [CRT_START], ax ; video page start offset
10493 00003DF5 AC                     <1>     lodsb
10494 00003DF6 A2[20680000]           <1>     mov     [CRT_COLS], al ; nbcols, characters per row
10495 00003DFB AC                     <1>     lodsb
10496 00003DFC A2[26680000]           <1>     mov     [VGA_ROWS], al ; nbrows, (character) rows per page (not rows-1)
10497 00003E01 AC                     <1>     lodsb
10498 00003E02 A2[22680000]           <1>     mov     [CHAR_HEIGHT], al ; character font height (8 or 16 or 14)
10499 00003E07 AC                     <1>     lodsb
10500 00003E08 A2[23680000]           <1>     mov     [VGA_VIDEO_CTL], al ; ROM BIOS DATA AREA Offset 87h
10501 00003E0D AC                     <1>     lodsb
10502 00003E0E A2[24680000]           <1>     mov     [VGA_SWITCHES], al ; feature bit switches
10503 00003E13 AC                     <1>     lodsb
10504 00003E14 A2[25680000]           <1>     mov     [VGA_MODESET_CTL], al ; basic mode set options
10505                                  <1>     ; followings are only used by TRDOS 386 v2 (IBM PC/AT ROMBIOS) code
10506                                  <1>     ; (bochs/plex86 does not use and return those)
10507 00003E19 AC                     <1>     lodsb
10508 00003E1A A2[21680000]           <1>     mov     [CRT_PALETTE], al ; current color palette ; TRDOS 386 feature only !
10509 00003E1F AC                     <1>     lodsb
10510 00003E20 A2[AE760100]           <1>     mov     [ACTIVE_PAGE], al ; current video page
10511 00003E25 66AD                   <1>     lodsw
10512 00003E27 66A3[37680000]         <1>     mov     [CURSOR_MODE], ax ; cursor type
10513                                  <1>     ;lodsd
10514                                  <1>     ;mov    [CURSOR_POSN], eax ; cursor position for video page 0 and 1
10515                                  <1>     ;lodsd
10516                                  <1>     ;mov    [CURSOR_POSN+4], eax ; cursor position for video page 2 and 3
10517                                  <1>     ;lodsd
10518                                  <1>     ;mov    [CURSOR_POSN+8], eax ; cursor position for video page 4 and 5
10519                                  <1>     ;lodsd
10520                                  <1>     ;mov    [CURSOR_POSN+12], eax ; cursor position for video page 6 and 7
10521 00003E2D B504                   <1>     mov     ch, 4
10522 00003E2F BF[9E760100]           <1>     mov     edi, CURSOR_POSN
10523                                  <1> bfn_rvs_8:
10524 00003E34 A5                     <1>     movsd
10525 00003E35 FECD                   <1>     dec     ch
10526 00003E37 75FB                   <1>     jnz     short bfn_rvs_8
10527                                  <1>     ; (font addr) protected mode address in kernel's/system memory space
10528                                  <1>     ; (not accessable/meaningful address value by user)
10529 00003E39 AD                     <1>     lodsd
10530 00003E3A A3[2A830100]           <1>     mov     [VGA_INT43H], eax ; VGA current (default) font address
10531                                  <1>
10532                                  <1>     ; (total 40 bytes are read&written above as BIOS data state)
10533                                  <1> bfn_rvs_9:
10534                                  <1>     ; 13/01/2021
10535 00003E3F F6C104                 <1>     test    cl, 4
10536 00003E42 7421                   <1>     jz      short bfn_rvs_11
10537                                  <1>
10538                                  <1>     ;BX++;
10539                                  <1>         ;v = read_byte(ES, BX); BX++;
10540                                  <1>         ;outb(VGAREG_PEL_MASK, read_byte(ES, BX)); BX++;
10541                                  <1>         ;// Set the whole dac always, from 0
10542                                  <1>         ;outb(VGAREG_DAC_WRITE_ADDRESS,0x00);
10543                                  <1>         ;for(i=0;i<256*3;i++) {
10544                                  <1>         ;   outb(VGAREG_DAC_DATA, read_byte(ES, BX)); BX++;
10545                                  <1>         ;}
10546                                  <1>         ;BX++;
10547                                  <1>         ;outb(VGAREG_DAC_WRITE_ADDRESS, v);
10548                                  <1>
10549                                  <1>     ; /* read/write mode dac */
10550 00003E44 AC                     <1>     lodsb   ; skip ; VGAREG_DAC_STATE
10551 00003E45 AC                     <1>     lodsb
10552 00003E46 88C4                   <1>     mov     ah, al ; * ; v
10553 00003E48 AC                     <1>     lodsb
10554 00003E49 66BAC603               <1>     mov     dx, VGAREG_PEL_MASK  ; 3C6h
10555 00003E4D EE                     <1>     out     dx, al
10556                                  <1>     ;// Set the whole dac always, from 0
10557 00003E4E 30C0                   <1>     xor     al, al ; 0
10558                                  <1>     ;mov    dx, VGAREG_DAC_WRITE_ADDRESS ; 3C8h
10559 00003E50 B2C8                   <1>     mov     dl, 0C8h
10560 00003E52 EE                     <1>     out     dx, al
10561                                  <1>
10562 00003E53 51                     <1>     push    ecx ; 22/01/2021
10563                                  <1>     ;for(i=0;i<256*3;i++) {
10564                                  <1>     ;mov    ecx, 256*3 ; 768 bytes
10565                                  <1>     ; 03/08/2022
10566 00003E54 29C9                   <1>     sub     ecx, ecx
10567 00003E56 B503                   <1>     mov     ch, 3
10568                                  <1>     ; ecx = 300h = 768
10569                                  <1>     ;mov    dx, VGAREG_DAC_DATA ; 3C9h
10570                                  <1>     ;mov    dl, 0C9h
10571 00003E58 FEC2                   <1>     inc     dl ; dx = 3C9h
10572                                  <1> bfn_rvs_10:
10573 00003E5A AC                     <1>     lodsb
10574 00003E5B EE                     <1>     out     dx, al
10575 00003E5C E2FC                   <1>     loop    bfn_rvs_10
10576 00003E5E 59                     <1>     pop     ecx ; 22/01/2021
10577                                  <1>
10578                                  <1>     ; /* color select register */
10579 00003E5F AC                     <1>     lodsb    ; skip
10580                                  <1>
10581 00003E60 88E0                   <1>     mov     al, ah ; * ; v
10582                                  <1>
10583                                  <1>     ;mov    dx, VGAREG_DAC_WRITE_ADDRESS ; 3C8h
10584                                  <1>     ;mov    dl, 0C8h
```

```
10585 00003E62 FECA                <1>     dec     dl ; dx  = 3C8h
10586 00003E64 EE                  <1>     out     dx, al ; * ; v
10587                              <1>
10588                              <1>     ; (total 772 bytes are read above as DAC state)
10589                              <1> bfn_rvs_11:
10590 00003E65 C3                  <1>     retn
10591                              <1>
10592                              <1> vbe_biosfn_restore_video_state:
10593                              <1>     ; 23/01/2021
10594                              <1>     ; 13/01/2021 (TRDOS 386 v2.0.3)
10595                              <1>     ; (vbe.c)
10596                              <1>
10597                              <1>     ; modified registers: eax, edx, esi, ch
10598                              <1>
10599                              <1>     ; input: esi = state buffer address
10600                              <1>     ; output:
10601                              <1>     ;       VBE DISPI register contents will be restored
10602                              <1>     ;       (18 bytes, 9 words)
10603                              <1>
10604                              <1>     ; enable = read_word(ES, BX);
10605                              <1>     ; BX += 2;
10606                              <1>     ;
10607                              <1>     ; if (!(enable & VBE_DISPI_ENABLED)) {
10608                              <1>         ;   outw(VBE_DISPI_IOPORT_INDEX,VBE_DISPI_INDEX_ENABLE);
10609                              <1>         ;   outw(VBE_DISPI_IOPORT_DATA, enable);
10610                              <1>     ; } else {
10611                              <1>         ;   outw(VBE_DISPI_IOPORT_INDEX, VBE_DISPI_INDEX_XRES);
10612                              <1>         ;   outw(VBE_DISPI_IOPORT_DATA, read_word(ES, BX));
10613                              <1>         ;   BX += 2;
10614                              <1>         ;   outw(VBE_DISPI_IOPORT_INDEX, VBE_DISPI_INDEX_YRES);
10615                              <1>         ;   outw(VBE_DISPI_IOPORT_DATA, read_word(ES, BX));
10616                              <1>         ;   BX += 2;
10617                              <1>         ;   outw(VBE_DISPI_IOPORT_INDEX, VBE_DISPI_INDEX_BPP);
10618                              <1>         ;   outw(VBE_DISPI_IOPORT_DATA, read_word(ES, BX));
10619                              <1>         ;   BX += 2;
10620                              <1>         ;   outw(VBE_DISPI_IOPORT_INDEX,VBE_DISPI_INDEX_ENABLE);
10621                              <1>         ;   outw(VBE_DISPI_IOPORT_DATA, enable);
10622                              <1>     ;
10623                              <1>         ;  for(i = VBE_DISPI_INDEX_BANK; i <= VBE_DISPI_INDEX_Y_OFFSET; i++)
10624                              <1>     ;    {
10625                              <1>         ;     outw(VBE_DISPI_IOPORT_INDEX, i);
10626                              <1>         ;     outw(VBE_DISPI_IOPORT_DATA, read_word(ES, BX));
10627                              <1>         ;     BX += 2;
10628                              <1>         ;    }
10629                              <1>         ; }
10630                              <1>
10631 00003E66 66AD                <1>     lodsw   ; enable (status, enabled=1, disabled=0)
10632 00003E68 66BACE01            <1>     mov     dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
10633                              <1>     ; 23/01/2021
10634 00003E6C 6683E001            <1>     and     ax, 1 ; VBE_DISPI_ENABLED
10635 00003E70 750B                <1>     jnz     short vbe_bfn_rvs_1
10636                              <1>     ; ax = 0
10637                              <1>     ; VBE_DISPI_DISABLED
10638                              <1> vbe_bfn_rvs_0:
10639                              <1>     ; enable (disable) dispi
10640                              <1>     ; dx = 01CEh ; VBE_DISPI_IOPORT_INDEX
10641                              <1>     ; ah = 0
10642 00003E72 50                  <1>     push    eax
10643 00003E73 B004                <1>     mov     al, 04h ; VBE_DISPI_INDEX_ENABLE
10644 00003E75 66EF                <1>     out     dx, ax
10645                              <1>     ;mov    dx, 01CFh ; VBE_DISPI_IOPORT_DATA
10646 00003E77 FEC2                <1>     inc     dl
10647 00003E79 58                  <1>     pop     eax
10648 00003E7A 66EF                <1>     out     dx, ax ; enable (or disable)
10649 00003E7C C3                  <1>     retn
10650                              <1> vbe_bfn_rvs_1:
10651                              <1>     ; VBE_DISPI_ENABLED
10652                              <1>
10653                              <1>     ; from VBE_DISPI_INDEX_XRES
10654                              <1>     ;   to VBE_DISPI_INDEX_BPP
10655                              <1>
10656 00003E7D B503                <1>     mov     ch, 3
10657 00003E7F 28C0                <1>     sub     al, al ; 0 ; VBE_DISPI_INDEX_XRES - 1
10658                              <1>     ; ax = 0
10659                              <1>
10660 00003E81 E80B000000          <1>     call    vbe_bfn_rvs_2
10661                              <1>
10662                              <1>         ;outw(VBE_DISPI_IOPORT_INDEX,VBE_DISPI_INDEX_ENABLE);
10663                              <1>     ;outw(VBE_DISPI_IOPORT_DATA, enable);
10664                              <1>
10665                              <1>     ; 23/01/2021
10666 00003E86 B001                <1>     mov     al, 1 ; VBE_DISPI_ENABLED
10667                              <1>     ; ax = 1
10668 00003E88 E8E5FFFFFF          <1>     call    vbe_bfn_rvs_0
10669                              <1>
10670                              <1>     ; from VBE_DISPI_INDEX_BANK
10671                              <1>     ;   to VBE_DISPI_INDEX_Y_OFFSET
10672                              <1>
10673 00003E8D B505                <1>     mov     ch, 5
10674                              <1>     ; 23/01/2021
10675 00003E8F B004                <1>     mov     al, 4  ; VBE_DISPI_INDEX_BANK - 1
10676                              <1>     ; ax = 4
10677                              <1> vbe_bfn_rvs_2:
10678 00003E91 FEC0                <1>     inc     al ; from VBE_DISPI_INDEX_XRES
10679                              <1>                    ; to VBE_DISPI_INDEX_BPP
10680                              <1>     ;mov    dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
10681                              <1>     ;mov    dl, 0CEh
10682 00003E93 66EF                <1>     out     dx, ax
10683 00003E95 50                  <1>     push    eax
10684                              <1>     ;mov    dx, 01CFh ; VBE_DISPI_IOPORT_DATA
10685 00003E96 FEC2                <1>     inc     dl ; 1CFh
10686 00003E98 66AD                <1>     lodsw
10687 00003E9A 66EF                <1>     out     dx, ax
10688 00003E9C 58                  <1>     pop     eax
10689 00003E9D FECA                <1>     dec     dl ; 1CEh
10690 00003E9F FECD                <1>     dec     ch
10691 00003EA1 75EE                <1>     jnz     short vbe_bfn_rvs_2
10692 00003EA3 C3                  <1>     retn
10693                              <1>
10694                              <1> ; ---------------------------------------------------------
10695                              <1>
10696                              <1> dispi_set_enable:
10697                              <1>     ; 03/08/2022
10698                              <1>     ; 23/11/2020
10699                              <1>     ; Input:
10700                              <1>     ;       ax = VBE_DISPI_ENABLED = 1
10701                              <1>     ;            or VBE_DISPI_DISABLED = 0
10702                              <1>     ;
10703                              <1>     ; Modified registers: none
10704                              <1>
10705                              <1>     ;push   edx
10706                              <1>     ;push   eax
10707                              <1>     ;mov    dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
10708                              <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
                                <1>     ;mov   dx, 04h ; VBE_DISPI_INDEX_ENABLE
                                <1>     ; 03/08/2022
00003EA4 28F6                   <1>     sub    dh, dh
00003EA6 B204                   <1>     mov    dl, 04h ; VBE_DISPI_INDEX_ENABLE
                                <1>
                                <1>     ;;call dispi_set_parms
                                <1>     ;;pop  edx
                                <1>     ;;retn
                                <1>     ;jmp   short dispi_set_parms
                                <1>
                                <1> dispi_set_parms:
                                <1>     ; 03/08/2022
                                <1>     ; 25/11/2020
                                <1>     ; Input:
                                <1>     ;      ax = data
                                <1>     ;      dx = vbe dispi register index
                                <1>     ;
                                <1>     ; Modified registers: edx
                                <1>
00003EA8 50                     <1>     push   eax
                                <1>     ;mov   ax, dx
                                <1>     ; 03/08/2022
00003EA9 89D0                   <1>     mov    eax, edx
00003EAB 66BACE01               <1>     mov    dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
00003EAF 66EF                   <1>     out    dx, ax
00003EB1 58                     <1>     pop    eax
                                <1>     ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
                                <1>     ;mov   dl, 0CFh
00003EB2 FEC2                   <1>     inc    dl
00003EB4 66EF                   <1>     out    dx, ax
00003EB6 C3                     <1>     retn
                                <1>
                                <1> dispi_set_bpp:
                                <1>     ; 03/08/2022
                                <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
                                <1>     ;mov   dx, 03h ; VBE_DISPI_INDEX_BPP
                                <1>     ; 03/08/2022
00003EB7 28F6                   <1>     sub    dh, dh
00003EB9 B203                   <1>     mov    dl, 03h ; VBE_DISPI_INDEX_BPP
                                <1>
                                <1>     ;;call dispi_set_parms
                                <1>     ;;pop  edx
                                <1>     ;;retn
00003EBB EBEB                   <1>     jmp    short dispi_set_parms
                                <1>
                                <1> dispi_set_xres:
                                <1>     ; 03/08/2022
                                <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
                                <1>     ;mov   dx, 01h ; VBE_DISPI_INDEX_XRES
                                <1>     ; 03/08/2022
00003EBD 28F6                   <1>     sub    dh, dh
00003EBF B201                   <1>     mov    dl, 01h ; VBE_DISPI_INDEX_XRES
                                <1>     ;;call dispi_set_parms
                                <1>     ;;pop  edx
                                <1>     ;;retn
00003EC1 EBE5                   <1>     jmp    short dispi_set_parms
                                <1>
                                <1> dispi_set_yres:
                                <1>     ; 03/08/2022
                                <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
                                <1>     ;mov    dx, 02h ; VBE_DISPI_INDEX_YRES
                                <1>     ; 03/08/2022
00003EC3 28F6                   <1>     sub     dh, dh
00003EC5 B202                   <1>     mov     dl, 02h ; VBE_DISPI_INDEX_YRES
                                <1>     ;;call  dispi_set_parms
                                <1>     ;;pop   edx
                                <1>     ;;retn
00003EC7 EBDF                   <1>     jmp     short dispi_set_parms
                                <1>
                                <1> dispi_set_bank:
                                <1>     ; 03/08/2022
                                <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
                                <1>     ;mov    dx, 05h ; VBE_DISPI_INDEX_BANK
                                <1>     ; 03/08/2022
00003EC9 28F6                   <1>     sub     dh, dh
00003ECB B205                   <1>     mov     dl, 05h ; VBE_DISPI_INDEX_BANK
                                <1>     ;;call  dispi_set_parms
                                <1>     ;;pop   edx
                                <1>     ;;retn
00003ECD EBD9                   <1>     jmp     short dispi_set_parms
                                <1>
                                <1> dispi_get_enable:
                                <1>     ; 03/08/2022
                                <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
                                <1>     ;mov    ax, 04h ; VBE_DISPI_INDEX_ENABLE
                                <1>     ; 03/08/2022
00003ECF 28E4                   <1>     sub     ah, ah
00003ED1 B004                   <1>     mov     al, 04h ; VBE_DISPI_INDEX_ENABLE
                                <1>
                                <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>
00003ED3 66BACE01               <1>     mov     dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
00003ED7 66EF                   <1>     out     dx, ax
                                <1>     ;mov    dx, 01CFh ; VBE_DISPI_IOPORT_DATA
                                <1>     ;mov    dl, 0CFh
00003ED9 FEC2                   <1>     inc     dl
00003EDB 66ED                   <1>     in      ax, dx
00003EDD C3                     <1>     retn
                                <1>
                                <1> vga_compat_setup:
                                <1>     ; 03/08/2022
                                <1>     ; 26/11/2020
                                <1>     ; 25/11/2020
                                <1>     ; VGA compatibility setup
                                <1>     ; (vbe.c, 02/01/2020, vruppert)
                                <1>     ;
                                <1>     ; Input:
                                <1>     ;       none
                                <1>     ;
                                <1>     ; Modified registers: eax, edx
                                <1>
                                <1>     ; 26/11/2020
                                <1>     ;push   eax
                                <1>     ;push   edx
                                <1>
                                <1>     ; set CRT X resolution
```

```
10957 00003EDE 66BACE01            <1>     mov     dx, 1CEh ; VBE_DISPI_IOPORT_INDEX
10958                               <1>     ;mov    ax, 01h  ; VBE_DISPI_INDEX_XRES
10959                               <1>     ; 03/08/2022
10960 00003EE2 31C0                <1>     xor     eax, eax
10961 00003EE4 FEC0                <1>     inc     al
10962                               <1>     ; eax = 1
10963 00003EE6 66EF                <1>     out     dx, ax
10964                               <1>     ;mov    dx, 1CFh ; VBE_DISPI_IOPORT_DATA
10965 00003EE8 FEC2                <1>     inc     dl
10966 00003EEA 66ED                <1>     in      ax, dx
10967 00003EEC 50                  <1>     push    eax
10968 00003EED 66BAD403            <1>     mov     dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
10969 00003EF1 66B81100            <1>     mov     ax, 0011h ; Vertical retrace end register
10970 00003EF5 66EF                <1>     out     dx, ax
10971                               <1>     ;pop    eax
10972                               <1>     ;push   eax
10973 00003EF7 8B0424              <1>     mov     eax, [esp]
10974                               <1>     ;shr    ax, 3 ; / 8 for pixel to character
10975                               <1>     ;dec    ax  ; - 1 (EGA or VGA?)
10976                               <1>     ; 03/08/2022
10977 00003EFA C1E803              <1>     shr     eax, 3
10978 00003EFD 48                  <1>     dec     eax
10979 00003EFE 88C4                <1>     mov     ah, al
10980 00003F00 B001                <1>     mov     al, 01h ; Horizontal display end register
10981 00003F02 66EF                <1>     out     dx, ax
10982 00003F04 58                  <1>     pop     eax
10983                               <1>
10984 00003F05 E89C000000          <1>     call    vga_set_virt_width
10985                               <1>
10986                               <1>     ; set CRT Y resolution
10987                               <1>     ; 03/08/2022
10988 00003F0A 66BACE01            <1>     mov     dx, 1CEh ; VBE_DISPI_IOPORT_INDEX
10989 00003F0E 66B80200            <1>     mov     ax, 02h  ; VBE_DISPI_INDEX_YRES
10990 00003F12 66EF                <1>     out     dx, ax
10991                               <1>     ;mov    dx, 1CFh ; VBE_DISPI_IOPORT_DATA
10992                               <1>     ; 03/08/2022
10993 00003F14 FEC2                <1>     inc     dl
10994 00003F16 66ED                <1>     in      ax, dx
10995 00003F18 50                  <1>     push    eax
10996 00003F19 66BAD403            <1>     mov     dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
10997 00003F1D 88C4                <1>     mov     ah, al
10998 00003F1F B012                <1>     mov     al, 12h ; Vertical display end register
10999 00003F21 66EF                <1>     out     dx, ax
11000 00003F23 58                  <1>     pop     eax
11001 00003F24 B007                <1>     mov     al, 07h ; Overflow register
11002 00003F26 EE                  <1>     out     dx, al
11003                               <1>     ;inc    dx
11004                               <1>     ; 03/08/2022
11005 00003F27 FEC2                <1>     inc     dl
11006 00003F29 EC                  <1>     in      al, dx ; read overflow register
11007 00003F2A 24BD                <1>     and     al, 0BDh ; clear VDE 9th and 10th bits
11008 00003F2C F6C401              <1>     test    ah, 01h
11009 00003F2F 7402                <1>     jz      short bit8_clear
11010 00003F31 0C02                <1>     or      al, 02h ; VDE 9th bit (bit 8) in bit 1
11011                               <1> bit8_clear:
11012 00003F33 F6C402              <1>     test    ah, 02h
11013 00003F36 7402                <1>     jz      short bit9_clear
11014 00003F38 0C40                <1>     or      al, 40h ; VDE 10th bit (bit 9) in bit 6
11015                               <1> bit9_clear:
11016 00003F3A EE                  <1>     out     dx, al
11017                               <1>
11018                               <1>     ; other settings
11019                               <1>     ;mov    dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
11020                               <1>     ; 03/08/2022
11021 00003F3B B2D4                <1>     mov     dl, 0D4h
11022 00003F3D 66B80900            <1>     mov     ax, 0009h ; Maximum scan line register
11023 00003F41 66EF                <1>     out     dx, ax  ; Reset
11024 00003F43 B017                <1>     mov     al, 17h ; Mode control register
11025 00003F45 EE                  <1>     out     dx, al
11026                               <1>     ;mov    dx, 3D5h ; VGAREG_VGA_CRTC_DATA
11027 00003F46 FEC2                <1>     inc     dl
11028 00003F48 EC                  <1>     in      al, dx  ; Read mode control register
11029 00003F49 0C03                <1>     or      al, 03h ; Set SRS and CMS bits
11030 00003F4B EE                  <1>     out     dx, al
11031                               <1>     ;mov    dx, 3DAh ; VGAREG_ACTL_RESET
11032                               <1>     ; 03/08/2022
11033 00003F4C B2DA                <1>     mov     dl, 0DAh
11034 00003F4E EC                  <1>     in      al, dx   ; clear flip-flop
11035                               <1>     ;mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
11036                               <1>     ; 03/08/2022
11037 00003F4F B2C0                <1>     mov     dl, 0C0h
11038 00003F51 B010                <1>     mov     al, 10h ; Mode control register
11039 00003F53 EE                  <1>     out     dx, al
11040                               <1>     ;mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
11041 00003F54 FEC2                <1>     inc     dl
11042 00003F56 EC                  <1>     in      al, dx
11043 00003F57 0C01                <1>     or      al, 01h ; select graphics mode
11044                               <1>     ;mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
11045 00003F59 FECA                <1>     dec     dl
11046 00003F5B EE                  <1>     out     dx, al ; Write to mode control register
11047 00003F5C B020                <1>     mov     al, 20h ; Palette RAM <-> display memory
11048 00003F5E EE                  <1>     out     dx, al ; Write to attribute addr register
11049                               <1>     ;mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
11050                               <1>     ; 03/08/2022
11051 00003F5F B2CE                <1>     mov     dl, 0CEh
11052 00003F61 66B80605            <1>     mov     ax, 0506h ; Misc. register, graph, mm 1
11053 00003F65 66EF                <1>     out     dx, ax
11054                               <1>     ;mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
11055                               <1>     ; 03/08/2022
11056 00003F67 B2C4                <1>     mov     dl, 0C4h
11057 00003F69 66B8020F            <1>     mov     ax, 0F02h ; Map mask register, all planes
11058 00003F6D 66EF                <1>     out     dx, ax
11059                               <1>
11060                               <1>     ; settings for >= 8bpp
11061                               <1>
11062                               <1>     ;mov    dx, 1CEh ; VBE_DISPI_IOPORT_INDEX
11063                               <1>     ;mov    ax, 03h ; VBE_DISPI_INDEX_BPP
11064                               <1>     ;out    dx, ax
11065                               <1>     ;;mov   dx, 1CFh ; VBE_DISPI_IOPORT_DATA
11066                               <1>     ;inc    dl
11067                               <1>     ;in     ax, dx
11068                               <1>     ;cmp    al, 08h  ; < 8 bits per pixel
11069                               <1>     ;jb     short vga_compat_end
11070                               <1>
11071                               <1>     ;mov    dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
11072                               <1>     ; 03/08/2022
11073 00003F6F B2D4                <1>     mov     dl, 0D4h
11074 00003F71 B014                <1>     mov     al, 14h  ; Underline location register
11075 00003F73 EE                  <1>     out     dx, al
11076                               <1>     ;mov    dx, 3D5h ; VGAREG_VGA_CRTC_DATA
11077 00003F74 FEC2                <1>     inc     dl
11078 00003F76 EC                  <1>     in      al, dx
11079 00003F77 0C40                <1>     or      al, 40h  ; enable double word mode
11080 00003F79 EE                  <1>     out     dx, al
```

```
                                <1>     ;mov    dx, 3DAh ; VGAREG_ACTL_RESET
                                <1>     ; 03/08/2022
00003F7A B2DA                   <1>     mov     dl, 0DAh
00003F7C EC                     <1>     in      al, dx   ; clear flip-flop
                                <1>     ;mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
                                <1>     ; 03/08/2022
00003F7D B2C0                   <1>     mov     dl, 0C0h
00003F7F B010                   <1>     mov     al, 10h  ; Mode control register
00003F81 EE                     <1>     out     dx, al
                                <1>     ;mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
00003F82 FEC2                   <1>     inc     dl
00003F84 EC                     <1>     in      al, dx
00003F85 0C40                   <1>     or      al, 40h  ; Pixel clock select is 1
                                <1>     ;mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
00003F87 FECA                   <1>     dec     dl
00003F89 EE                     <1>     out     dx, al   ; update mode control reggister
00003F8A B020                   <1>     mov     al, 20h  ; select display memory as PAS
00003F8C EE                     <1>     out     dx, al
                                <1>     ;mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
                                <1>     ; 03/08/2022
00003F8D B2C4                   <1>     mov     dl, 0C4h
00003F8F B004                   <1>     mov     al, 04h ; Memory mode register
00003F91 EE                     <1>     out     dx, al
                                <1>     ;mov    dx, 3C5h ; VGAREG_SEQU_DATA
00003F92 FEC2                   <1>     inc     dl
00003F94 EC                     <1>     in      al, dx
00003F95 0C08                   <1>     or      al, 08h ; enable chain four
00003F97 EE                     <1>     out     dx, al
                                <1>     ;mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
                                <1>     ; 03/08/2022
00003F98 B2CE                   <1>     mov     dl, 0CEh
00003F9A B005                   <1>     mov     al, 05h  ; Mode register
00003F9C EE                     <1>     out     dx, al
                                <1>     ;mov    dx, 3CFh ; VGAREG_GRDC_DATA
00003F9D FEC2                   <1>     inc     dl
00003F9F EC                     <1>     in      al, dx
00003FA0 249F                   <1>     and     al, 9Fh  ; clear shift register
00003FA2 0C40                   <1>     or      al, 40h  ; set shift register to 2
00003FA4 EE                     <1>     out     dx, al
                                <1>
                                <1> vga_compat_end:
                                <1>     ;pop    edx
                                <1>     ;pop    eax
00003FA5 C3                     <1>     retn
                                <1>
                                <1> vga_set_virt_width:
                                <1>     ; 03/08/2022
                                <1>     ; 27/11/2020
                                <1>     ; 25/11/2020
                                <1>     ; (vbe.c, 02/01/2020, vruppert)
                                <1>     ;
                                <1>     ; Input:
                                <1>     ;       AX = resolution (screen width)
                                <1>     ;
                                <1>     ; Modified registers: eax, edx
                                <1>
                                <1>     ;;push  ebx
                                <1>     ;push   edx
                                <1>     ;push   eax
                                <1>     ;mov    ebx, eax
                                <1>     ;call   dispi_get_bpp ; bits per pixel
                                <1>     ;cmp    al, 4
                                <1>     ;ja     short set_width_svga  ; 8, 16, 24, 32
                                <1>     ;shr    bx, 1
                                <1> ;set_width_svga:
                                <1>     ;shr    bx, 3
                                <1>     ;mov    eax, [esp]
                                <1>     ;shr    ax, 3   ; / 8, bytes per row
                                <1>     ; 03/08/2022
00003FA6 C1E803                 <1>     shr     eax, 3
00003FA9 66BAD403               <1>     mov     dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
                                <1>     ;mov    ah, bl  ;
00003FAD 88C4                   <1>     mov     ah, al  ; width in bytes
00003FAF B013                   <1>     mov     al, 13h ; offset register
00003FB1 66EF                   <1>     out     dx, ax  ; index (3D4h) and data (3D5h)
                                <1>     ;pop    eax
                                <1>     ;pop    edx
                                <1>     ;;pop   ebx
00003FB3 C3                     <1>     retn
                                <1>
                                <1> ; 24/11/2020
                                <1>
                                <1> struc bmi ; BOCHS/PLEX86 MODE INFO structure/table
00000000 ????                   <1>  .mode:       resw 1
00000002 ????                   <1>  .width:  resw 1
00000004 ????                   <1>  .height: resw 1
00000006 ????                   <1>  .depth:  resw 1
                                <1>  .size:
                                <1> endstruc
                                <1>
                                <1> ; 24/11/2020
                                <1> struc MODEINFO
00000000 ????                   <1>  .mode:                     resw 1  ; 1xxh
00000002 ????                   <1>  .ModeAttributes:  resw 1
00000004 ??                     <1>  .WinAAttributes:  resb 1
00000005 ??                     <1>  .WinBAttributes:  resb 1 ; = 0
00000006 ????                   <1>  .WinGranularity:  resw 1
00000008 ????                   <1>  .WinSize:         resw 1
0000000A ????                   <1>  .WinASegment:             resw 1
0000000C ????                   <1>  .WinBSegment:             resw 1 ; = 0
0000000E ????????               <1>  .WinFuncPtr:              resd 1 ; = 0
00000012 ????                   <1>  .BytesPerScanLine:        resw 1
00000014 ????                   <1>  .XResolution:             resw 1
00000016 ????                   <1>  .YResolution:             resw 1
00000018 ??                     <1>  .XCharSize:               resb 1
00000019 ??                     <1>  .YCharSize:               resb 1
0000001A ??                     <1>  .NumberOfPlanes:  resb 1
0000001B ??                     <1>  .BitsPerPixel:            resb 1
0000001C ??                     <1>  .NumberOfBanks:   resb 1
0000001D ??                     <1>  .MemoryModel:             resb 1
0000001E ??                     <1>  .BankSize:        resb 1 ; = 0
0000001F ??                     <1>  .NumberOfImagePages:      resb 1
00000020 ??                     <1>  .Reserved_page:   resb 1 ; = 0
00000021 ??                     <1>  .RedMaskSize:             resb 1
00000022 ??                     <1>  .RedFieldPosition:        resb 1
00000023 ??                     <1>  .GreenMaskSize:   resb 1
00000024 ??                     <1>  .GreenFieldPosition:      resb 1
00000025 ??                     <1>  .BlueMaskSize:            resb 1
00000026 ??                     <1>  .BlueFieldPosition:       resb 1
00000027 ??                     <1>  .RsvdMaskSize:            resb 1
00000028 ??                     <1>  .RsvdFieldPosition:       resb 1
00000029 ??                     <1>  .DirectColorModeInfo:     resb 1
0000002A ????????               <1>  .PhysBasePtr:             resd 1
0000002E ????????               <1>  .OffScreenMemOffset:      resd 1 ; = 0
```

```
11205 00000032 ????              <1>  .OffScreenMemSize:       resw 1  ; = 0
11206 00000034 ????              <1>  .LinBytesPerScanLine:    resw 1
11207 00000036 ??                <1>  .BnkNumberOfPages:       resb 1
11208 00000037 ??                <1>  .LinNumberOfPages:       resb 1
11209 00000038 ??                <1>  .LinRedMaskSize:  resb 1
11210 00000039 ??                <1>  .LinRedFieldPosition1: resb 1
11211 0000003A ??                <1>  .LinGreenMaskSize1:      resb 1
11212 0000003B ??                <1>  .LinGreenFieldPosition:resb 1
11213 0000003C ??                <1>  .LinBlueMaskSize:        resb 1
11214 0000003D ??                <1>  .LinBlueFieldPosition:   resb 1
11215 0000003E ??                <1>  .LinRsvdMaskSize:        resb 1
11216 0000003F ??                <1>  .LinRsvdFieldPosition:   resb 1
11217 00000040 ????????          <1>  .MaxPixelClock:   resd 1  ; = 0
11218                            <1> .size:
11219                            <1> endstruc
11220                            <1>
11221                            <1> ; 10/12/2020
11222                            <1> struc LFBINFO
11223 00000000 ????              <1>  .mode:                   resw 1  ; 1XXh
11224 00000002 ????????          <1>  .LFB_addr:        resd 1
11225 00000006 ????????          <1>  .LFB_size:        resd 1
11226 0000000A ????              <1>  .X_res:           resw 1
11227 0000000C ????              <1>  .Y_res:           resw 1
11228 0000000E ??                <1>  .bpp:                    resb 1
11229 0000000F ??                <1>  .reserved:        resb 1
11230                            <1> .size:      ; 16 bytes
11231                            <1> endstruc
11232                            <1>
11233                            <1> set_mode_info_list:
11234                            <1>     ; 14/12/2020
11235                            <1>     ; 11/12/2020
11236                            <1>     ; 24/11/2020
11237                            <1>     ; (vbetables-gen.c)
11238                            <1>     ; Input:
11239                            <1>     ;      BX = VBE mode (including bochs special modes)
11240                            <1>     ; Output:
11241                            <1>     ;      ;;EAX = MODE_INFO_LIST address
11242                            <1>     ;      EAX = 0 ; 11/12/2020
11243                            <1>     ;      ESI = MODE_INFO_LIST address ; 11/12/2020
11244                            <1>     ;      (if mode is not found, ESI = 0)
11245                            <1>     ;
11246                            <1>     ; Modified registers: eax, ebx, ecx, edx, esi, edi
11247                            <1>
11248 00003FB4 BE[AA6B0000]      <1>     mov    esi, b_vbe_modes ; bochs mode info base table
11249 00003FB9 BF[0A9D0100]      <1>     mov    edi, MODE_INFO_LIST ; mode info list (4F01h)
11250                            <1> sml_0:
11251 00003FBE 66AD              <1>     lodsw
11252 00003FC0 6639D8            <1>     cmp    ax, bx ; is mode number same ?
11253 00003FC3 7410              <1>     je     short sml_1 ; yes
11254 00003FC5 AD                <1>     lodsd
11255 00003FC6 66AD              <1>     lodsw
11256 00003FC8 81FE[6A6C0000]    <1>     cmp    esi, end_of_b_vbe_modes
11257 00003FCE 72EE              <1>     jb     short sml_0
11258                            <1>     ; not found
11259 00003FD0 31C0              <1>     xor    eax, eax ; 0
11260                            <1>     ; 11/12/2020
11261 00003FD2 31F6              <1>     xor    esi, esi
11262 00003FD4 C3                <1>     retn
11263                            <1> sml_1:
11264 00003FD5 66AB              <1>     stosw  ; mode
11265 00003FD7 AD                <1>     lodsd  ; width, height
11266                            <1>     ; 14/12/2020
11267 00003FD8 89C1              <1>     mov    ecx, eax
11268 00003FDA 50                <1>     push   eax ; ***
11269 00003FDB 29C0              <1>     sub    eax, eax ; clear high word of eax
11270 00003FDD 66AD              <1>     lodsw  ; depth
11271 00003FDF 50                <1>     push   eax ; **
11272                            <1>
11273                            <1>     ;add   al, 7 ; only for 15 bit colors (not used here)
11274 00003FE0 C0E803            <1>     shr    al, 3 ; / 8
11275                            <1>     ; 14/12/2020
11276 00003FE3 66F7E1            <1>     mul    cx  ; pitch = width * ((depth+7)/8)
11277                            <1>            ; ax = pitch
11278 00003FE6 50                <1>     push   eax ; * ; high word of eax = 0
11279 00003FE7 C1E910            <1>     shr    ecx, 16
11280                            <1>     ;mul   cx
11281                            <1>     ;mov   cx, ax
11282 00003FEA 31D2              <1>     xor    edx, edx  ; clear high word of edx
11283 00003FEC F7E1              <1>     mul    ecx ; height * pitch
11284 00003FEE 89C1              <1>     mov    ecx, eax
11285 00003FF0 B800000001        <1>     mov    eax, VBE_DISPI_TOTAL_VIDEO_MEMORY_MB * 1024 * 1024
11286 00003FF5 F7F1              <1>     div    ecx
11287                            <1>            ; eax = pages = vram_size / (height*pitch)
11288                            <1>
11289                            <1>     ;mov   cx, ax
11290 00003FF7 89C1              <1>     mov    ecx, eax ; pages
11291                            <1>
11292 00003FF9 66B89B00          <1>     mov    ax, MODE_ATTRIBUTES
11293 00003FFD 66AB              <1>     stosw  ; ModeAttributes
11294 00003FFF B007              <1>     mov    al, WINA_ATTRIBUTES
11295 00004001 AA                <1>     stosb  ; WinAAttributes
11296 00004002 30C0              <1>     xor    al, al ; WinBAttributes = 0
11297 00004004 AA                <1>     stosb
11298 00004005 66B84000          <1>     mov    ax, VBE_DISPI_BANK_SIZE_KB
11299 00004009 66AB              <1>     stosw  ; WinGranularity
11300 0000400B 66AB              <1>     stosw  ; WinSize
11301 0000400D 66B800A0          <1>     mov    ax, VGAMEM_GRAPH
11302 00004011 66AB              <1>     stosw  ; WinASegment
11303 00004013 29C0              <1>     sub    eax, eax
11304 00004015 66AB              <1>     stosw  ; WinBSegment = 0
11305 00004017 AB                <1>     stosd  ; WinFuncPtr = 0
11306                            <1>
11307 00004018 58                <1>     pop    eax ; * ; pitch
11308 00004019 89C3              <1>     mov    ebx, eax  ; high word of ebx = 0 ; 14/12/2020
11309 0000401B 66AB              <1>     stosw  ; BytesPerScanLine
11310                            <1>
11311 0000401D 5A                <1>     pop    edx ; ** ; depth (bits per pixel)
11312 0000401E 58                <1>     pop    eax ; *** width, height
11313                            <1>
11314                            <1>     ; // Mandatory information for VBE 1.2 and above
11315                            <1>
11316 0000401F 66AB              <1>     stosw  ; XResolution (width)
11317 00004021 C1E810            <1>     shr    eax, 16
11318 00004024 50                <1>     push   eax ; **** height
11319 00004025 66AB              <1>     stosw  ; YResolution (height)
11320 00004027 B008              <1>     mov    al, 8
11321 00004029 AA                <1>     stosb  ; XCharSize  ; char width
11322 0000402A B010              <1>     mov    al, 16
11323 0000402C AA                <1>     stosb  ; YCharSize  ; char height
11324 0000402D B001              <1>     mov    al, 1
11325 0000402F AA                <1>     stosb  ; NumberOfPlanes
11326                            <1>     ;movzx eax, dl
11327 00004030 88D0              <1>     mov    al, dl ; eax <= 32
11328 00004032 AA                <1>     stosb  ; BitsPerPixel
```

```
11329                                   <1>     ; Number of banks = (height * pitch + 65535) / 65536
11330 00004033 58                       <1>     pop     eax ; **** ; height
11331                                   <1>     ; 14/12/2020
11332 00004034 52                       <1>     push    edx ; ***** ; depth ; edx <= 32
11333 00004035 F7E3                     <1>     mul     ebx  ; pitch (ebx) * height (eax)
11334                                   <1>     ;mov    edx, [esp] ; *****
11335                                   <1>     ;mov    dl, [esp] ; *****
11336 00004037 05FFFF0000               <1>     add     eax, 65535
11337 0000403C C1E810                   <1>     shr     eax, 16 ; / 65536 ; <= 127 ; 14/12/2020
11338 0000403F AA                       <1>     stosb   ; NumberOfBanks
11339                                   <1>     ; 14/12/2020
11340                                   <1>     ;cmp    dl, 8 ; 8 bits per pixel
11341 00004040 803C2408                 <1>     cmp     byte [esp], 8
11342 00004044 7704                     <1>     ja      short sml_2
11343 00004046 B004                     <1>     mov     al, VBE_MEMORYMODEL_PACKED_PIXEL
11344 00004048 EB02                     <1>     jmp     short sml_3
11345                                   <1> sml_2:
11346                                   <1>     ; 16, 24, 32 bits per pixel
11347 0000404A B006                     <1>     mov     al, VBE_MEMORYMODEL_DIRECT_COLOR
11348                                   <1> sml_3:
11349 0000404C AA                       <1>     stosb
11350 0000404D 30C0                     <1>     xor     al, al ; 0
11351 0000404F AA                       <1>     stosb   ; BankSize = 0
11352 00004050 49                       <1>     dec     ecx ; pages - 1
11353                                   <1>     ; NumberOfImagePages = 262 for 320x200x8 mode
11354                                   <1>     ;;mov   ax, 255
11355                                   <1>     ; 14/12/2020
11356                                   <1>     ;mov    al, 255
11357 00004051 FEC8                     <1>     dec     al ; 255
11358 00004053 39C1                     <1>     cmp     ecx, eax ; ecx <= 261, eax = 255
11359                                   <1>     ;cmp    cx, ax
11360 00004055 7302                     <1>     jnb     short sml_4
11361 00004057 88C8                     <1>     mov     al, cl
11362                                   <1> sml_4:
11363 00004059 AA                       <1>     stosb   ; NumberOfImagePages (1 byte)
11364 0000405A 28C0                     <1>     sub     al, al
11365 0000405C AA                       <1>     stosb   ; Reserved_page = 0
11366 0000405D 58                       <1>     pop     eax ; ***** ; depth
11367 0000405E 88C1                     <1>     mov     cl, al
11368                                   <1>     ; eax < = 32
11369 00004060 2C08                     <1>     sub     al, 8 ; 8->0, 16->8, 24->16, 32->24
11370 00004062 BE[6A6C0000]             <1>     mov     esi, direct_color_fields
11371 00004067 01C6                     <1>     add     esi, eax
11372 00004069 56                       <1>     push    esi ; ******
11373 0000406A AD                       <1>     lodsd   ; RedMaskSize (AL), RedFieldPosition (AH)
11374                                   <1>             ; GreenMaskSize (16), GreenFieldPosition (24)
11375 0000406B AB                       <1>     stosd
11376 0000406C AD                       <1>     lodsd   ; BlueMaskSize (AL), BlueFieldPosition (AH)
11377                                   <1>             ; RsvdMaskSize (16), RsvdFieldPosition (24)
11378 0000406D AB                       <1>     stosd
11379 0000406E 5E                       <1>     pop     esi ; ******
11380                                   <1>
11381 0000406F 30C0                     <1>     xor     al, al ; 0
11382 00004071 80F920                   <1>     cmp     cl, 32
11383 00004074 7202                     <1>     jb      short sml_5
11384 00004076 B002                     <1>     mov     al, VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
11385                                   <1> sml_5:
11386 00004078 AA                       <1>     stosb   ; DirectColorModeInfo
11387                                   <1>
11388                                   <1>     ; // Mandatory information for VBE 2.0 and above
11389                                   <1>
11390 00004079 B8000000E0               <1>     mov     eax, VBE_DISPI_LFB_PHYSICAL_ADDRESS
11391 0000407E AB                       <1>     stosd   ; PhysBasePtr
11392 0000407F 29C0                     <1>     sub     eax, eax
11393 00004081 AB                       <1>     stosd   ; OffScreenMemOffset = 0
11394 00004082 66AB                     <1>     stosw   ; OffScreenMemSize = 0
11395                                   <1>
11396                                   <1>     ;// Mandatory information for VBE 3.0 and above
11397                                   <1>
11398                                   <1>     ; ebx = pitch
11399 00004084 6689D8                   <1>     mov     ax, bx
11400                                   <1>     ;stosw
11401                                   <1>
11402                                   <1>     ;xor    al, al
11403                                   <1>             ;stosb  ; BnkNumberOfPages = 0
11404                                   <1>             ;stosb  ; LinNumberOfPages = 0
11405                                   <1>
11406 00004087 AB                       <1>     stosd   ; pitch (word), 0 (byte), 0 (byte)
11407                                   <1>
11408 00004088 AD                       <1>     lodsd   ; LinRedMaskSize (AL), LinRedFieldPosition (AH)
11409                                   <1>             ; LinGreenMaskSize (16), LinGreenFieldPosition (24)
11410 00004089 AB                       <1>     stosd
11411 0000408A AD                       <1>     lodsd   ; LinBlueMaskSize (AL), LinBlueFieldPosition (AH)
11412                                   <1>             ; LinRsvdMaskSize (16), LinRsvdFieldPosition (24)
11413 0000408B AB                       <1>     stosd
11414                                   <1>
11415 0000408C 29C0                     <1>     sub     eax, eax
11416 0000408E AB                       <1>     stosd   ; MaxPixelClock = 0
11417                                   <1>
11418                                   <1>     ;mov    eax, MODE_INFO_LIST
11419                                   <1>     ; 11/12/2020
11420 0000408F BE[0A9D0100]             <1>     mov     esi, MODE_INFO_LIST
11421                                   <1>
11422 00004094 C3                       <1>     retn
11423                                   <1>
11424                                   <1> ; end of set_mode_info_list ; edi = set_mode_info_list + 68
11425                                   <1>
11426                                   <1> pci_get_lfb_addr:
11427                                   <1>     ; 11/12/2020
11428                                   <1>     ; Get linear frame buffer base from PCI
11429                                   <1>     ; (vgabios.c, 02/01/2020, vruppert)
11430                                   <1>     ;
11431                                   <1>     ; Input:
11432                                   <1>     ;       ax = PCI device vendor id
11433                                   <1>     ; Output:
11434                                   <1>     ;       ax  = LFB address (high 16 bit) (zf=0)
11435                                   <1>     ;       eax = 0 -> not found (error) (zf=1)
11436                                   <1>     ;
11437                                   <1>     ; Modified registers: eax
11438                                   <1>
11439 00004095 53                       <1>     push    ebx
11440 00004096 51                       <1>     push    ecx
11441 00004097 52                       <1>     push    edx
11442                                   <1>     ;
11443 00004098 89C3                     <1>     mov     ebx, eax
11444 0000409A 31C9                     <1>     xor     ecx, ecx
11445 0000409C 28D2                     <1>     sub     dl, dl  ; mov dl, 0
11446 0000409E E842000000               <1>     call    pci_read_reg
11447 000040A3 6683F8FF                 <1>     cmp     ax, 0FFFFh
11448 000040A7 7417                     <1>     je      short pci_get_lfb_addr_fail
11449                                   <1> pci_get_lfb_addr_next_dev:
11450 000040A9 28D2                     <1>     sub     dl, dl ; mov dl, 0
11451 000040AB E835000000               <1>     call    pci_read_reg
11452 000040B0 6639D8                   <1>     cmp     ax, bx  ; check vendor
```

```
11453 000040B3 740F                <1>     je      short pci_get_lfb_addr_found
11454 000040B5 6683C108            <1>     add     cx, 08h
11455 000040B9 6681F90002          <1>     cmp     cx, 200h ; search bus 0 and 1
11456 000040BE 72E9                <1>     jb      short pci_get_lfb_addr_next_dev
11457                              <1> pci_get_lfb_addr_fail:
11458 000040C0 31C0                <1>     xor     eax, eax ; no LFB
11459                              <1>     ; zf = 1
11460 000040C2 EB1D                <1>     jmp     short pci_get_lfb_addr_return
11461                              <1> pci_get_lfb_addr_found:
11462 000040C4 B210                <1>     mov     dl, 10h ; I/O space 0
11463 000040C6 E81A000000          <1>     call    pci_read_reg
11464 000040CB 66A9F1FF            <1>     test    ax, 0FFF1h
11465 000040CF 740D                <1>     jz      short pci_get_lfb_addr_success
11466 000040D1 B214                <1>     mov     dl, 14h ; I/O space 1
11467 000040D3 E80D000000          <1>     call    pci_read_reg
11468 000040D8 66A9F1FF            <1>     test    ax, 0FFF1h
11469 000040DC 75E2                <1>     jnz     short pci_get_lfb_addr_fail
11470                              <1> pci_get_lfb_addr_success:
11471 000040DE C1E810              <1>     shr     eax, 16 ; LFB address (hw)
11472                              <1>     ; zf = 0
11473                              <1> pci_get_lfb_addr_return:
11474 000040E1 5A                  <1>     pop     edx
11475 000040E2 59                  <1>     pop     ecx
11476 000040E3 5B                  <1>     pop     ebx
11477 000040E4 C3                  <1>     retn
11478                              <1>
11479                              <1> pci_read_reg:
11480                              <1>     ; 11/12/2020
11481                              <1>     ; Read PCI register
11482                              <1>     ; (vgabios.c, 02/01/2020, vruppert)
11483                              <1>     ;
11484                              <1>     ; Input:
11485                              <1>     ;       cx = device/function
11486                              <1>     ;       dl = register
11487                              <1>     ; Output:
11488                              <1>     ;       eax = value
11489                              <1>     ;
11490                              <1>     ; Modified registers: eax, edx
11491                              <1>
11492 000040E5 B800008000          <1>     mov     eax, 00800000h
11493 000040EA 6689C8              <1>     mov     ax, cx
11494 000040ED C1E008              <1>     shl     eax, 8
11495 000040F0 88D0                <1>     mov     al, dl
11496 000040F2 66BAF80C            <1>     mov     dx, 0CF8h
11497 000040F6 EF                  <1>     out     dx, eax
11498 000040F7 80C204              <1>     add     dl, 4 ; mov dx, 0CFCh
11499 000040FA ED                  <1>     in      eax, dx
11500 000040FB C3                  <1>     retn
11501                              <1>
11502                              <1> %endif
11503                              <1>
11504                              <1> ; -----------
11505                              <1>
11506                              <1> %if 0
11507                              <1>
11508                              <1> mode_info_find_mode:
11509                              <1>     ; 25/11/2020
11510                              <1>     ; Input:
11511                              <1>     ;       bx = VESA VBE2 video mode (+ bochs extensions)
11512                              <1>     ; Output:
11513                              <1>     ;       esi = mode info address (for BX input)
11514                              <1>     ;       esi = 0 -> not found
11515                              <1>     ;
11516                              <1>     ; Modified registers: eax, esi
11517                              <1>
11518                              <1>     xor     eax, eax
11519                              <1>     mov     esi, MODE_INFO_LIST
11520                              <1> mifm_1:
11521                              <1>     mov     ax, [esi]
11522                              <1>     cmp     ax, bx
11523                              <1>     je      short mifm_2
11524                              <1>     add     esi, MODEINFO.size ; add esi, 68
11525                              <1>     cmp     esi, VBE_VESA_MODE_END_OF_LIST
11526                              <1>     jb      short mifm_1
11527                              <1>     ; not found
11528                              <1>     sub     esi, esi ; 0
11529                              <1> mifm_2
11530                              <1>     retn
11531                              <1>
11532                              <1> dispi_get_bpp:
11533                              <1>     ; 28/11/2020
11534                              <1>     ; Input:
11535                              <1>     ;       none
11536                              <1>     ; Output:
11537                              <1>     ;       al = Bits per pixel
11538                              <1>     ;            (8,16,24,32)
11539                              <1>     ;       ah = Bytes per pixel
11540                              <1>     ;            (1,2,3,4)
11541                              <1>     ;
11542                              <1>     ; Modified registers: none
11543                              <1>
11544                              <1>     ;push   edx
11545                              <1>     ;mov    dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
11546                              <1>     ;mov    ax, 03h   ; VBE_DISPI_INDEX_BPP
11547                              <1>     ;out    dx, ax
11548                              <1>     ;;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
11549                              <1>     ;;mov   dl, 0CFh
11550                              <1>     ;inc    dl
11551                              <1>     ;in     ax, dx
11552                              <1>     ;mov    ah, al
11553                              <1>     ;shr    ah, 3 ; / 8
11554                              <1>     ;;test  al, 7 ; 15 bit graphics mode
11555                              <1>     ;;jz    short get_bpp_noinc
11556                              <1>     ;;inc   ah
11557                              <1> ;;get_bpp_noinc:
11558                              <1>     ;pop    edx
11559                              <1>     ;retn
11560                              <1>
11561                              <1>     ; 28/11/2020
11562                              <1>     ; Modified registers: edx
11563                              <1>     ;push   edx
11564                              <1>     mov     dx, 03h ; VBE_DISPI_INDEX_BPP
11565                              <1>     call    dispi_get_parms
11566                              <1>     ;pop    edx
11567                              <1>     ;retn
11568                              <1>     mov     ah, al
11569                              <1>     shr     ah, 3 ; / 8
11570                              <1>     ;test   al, 7 ; 15 bit graphips mode
11571                              <1>     ;jz     short get_bpp_noinc
11572                              <1>     ;inc    ah
11573                              <1> ;get_bpp_noinc:
11574                              <1>     ;pop    edx
11575                              <1>     retn
11576                              <1>
```

```
11577                           <1> restore_vesa_video_state:
11578                           <1>     ; 02/08/2022 (TRDOS 386 v2.0.5)
11579                           <1>     ; 14/01/2021
11580                           <1>     ; 06/12/2020
11581                           <1>     ; Input:
11582                           <1>     ;       [vbe3stbufsize] <= 32 ; <= 32*64 bytes
11583                           <1>     ; Output:
11584                           <1>     ;       AX = 004Fh (successed)
11585                           <1>     ;
11586                           <1>     ;       eax = 0 -> buffer size problem
11587                           <1>     ;       eax > 0 and ax <> 004Fh -> failed
11588                           <1>     ;
11589                           <1>     ; Modified regs: eax, ebx, ecx, edx, esi, edi
11590                           <1>
11591                           <1>     ;movzx  ecx, word [vbe3stbufsize]
11592                           <1>     ;cmp    cx, 32 ; 32 * 64 bytes
11593                           <1>     ;ja     short r_v_b_s_fail
11594                           <1>
11595                           <1>     movzx   ecx, byte [vbe3stbufsize]; <=32
11596                           <1>     ;shl    cx, 4 ; dword count for movsd
11597                           <1>     ; 02/08/2022
11598                           <1>     shl     ecx, 4 ; <= 32*16 dwords (32*64 bytes)
11599                           <1>     mov     edi, VBE3SAVERESTOREBLOCK ; destination
11600                           <1>                                       ; (vbe3 pmi buff)
11601                           <1>     mov     esi, VBE3VIDEOSTATE ; source (kernel buff)
11602                           <1>     rep     movsd
11603                           <1>
11604                           <1>     mov     ax, 4F04h
11605                           <1>     mov     dl, 02h ; restore
11606                           <1>     ;mov    cx, 0Fh
11607                           <1>     mov     cl, 0Fh
11608                           <1>     xor     ebx, ebx ; points to VBE3SAVERESTOREBLOCK
11609                           <1>     jmp     short int10h_32bit_pmi
11610                           <1>
11611                           <1> ;s_v_b_s_fail:
11612                           <1> ;r_v_b_s_fail:
11613                           <1> ;   xor     eax, eax
11614                           <1> ;   retn
11615                           <1>
11616                           <1> save_vesa_video_state:
11617                           <1>     ; 02/08/2022 (TRDOS 386 v2.0.5)
11618                           <1>     ; 14/01/2021
11619                           <1>     ; 06/12/2020
11620                           <1>     ; Input:
11621                           <1>     ;       [vbe3stbufsize] <= 32 ; <= 32*64 bytes
11622                           <1>     ; Output:
11623                           <1>     ;       AX = 004Fh (successed)
11624                           <1>     ;
11625                           <1>     ;       eax = 0 -> buffer size problem
11626                           <1>     ;       eax > 0 and ax <> 004Fh -> failed
11627                           <1>     ;
11628                           <1>     ; Modified regs: eax, ebx, ecx, edx, esi, edi
11629                           <1>
11630                           <1>     ;cmp    word [vbe33stbufsize], 32
11631                           <1>     ;                       ; 32 * 64 bytes
11632                           <1>     ;ja     short s_v_b_s_fail
11633                           <1>
11634                           <1>     mov     ax, 4F04h
11635                           <1>     mov     dl, 01h ; save
11636                           <1>     ;mov    cx, 0Fh
11637                           <1>     mov     cl, 0Fh
11638                           <1>     xor     ebx, ebx ; points to VBE3SAVERESTOREBLOCK
11639                           <1>
11640                           <1>     call    int10h_32bit_pmi
11641                           <1>
11642                           <1>     movzx   ecx, byte [vbe3stbufsize]; <=32
11643                           <1>     ;shl    cx, 4 ; dword count for movsd
11644                           <1>     ; 02/08/2022
11645                           <1>     shl     ecx, 4 ; <= 32*16 dwords (32*64 bytes)
11646                           <1>     mov     esi, VBE3SAVERESTOREBLOCK ; destination
11647                           <1>                                       ; (vbe3 pmi buff)
11648                           <1>     mov     edi, VBE3VIDEOSTATE ; source (kernel buff)
11649                           <1>     rep     movsd
11650                           <1>     retn
11651                           <1>
11652                           <1> dispi_set_bank_farcall:
11653                           <1>     ; 03/08/2022
11654                           <1>     ; 12/04/2021 (32 bit push/pop)
11655                           <1>     ; 11/12/2020
11656                           <1>     ; (This may be 'sysvideo' function, later)
11657                           <1>     ;
11658                           <1>     ; Input:
11659                           <1>     ;       bx = 0000h, set bank number
11660                           <1>     ;          = 0100h, get bank number
11661                           <1>     ;       dx = bank number (if bx = 0)
11662                           <1>     ; Output:
11663                           <1>     ;       dx = bank number
11664                           <1>
11665                           <1>     cmp     bx, 0100h
11666                           <1>     je      short dispi_set_bank_farcall_get
11667                           <1>     or      bx, bx
11668                           <1>     ;jnz    dispi_set_bank_farcall_error
11669                           <1>     ; 03/08/2022
11670                           <1>     jz      short dsbfcall_1
11671                           <1>     jmp     dispi_set_bank_farcall_error
11672                           <1> dsbfcall_1:
11673                           <1>     mov     ax, dx
11674                           <1>     ;push   dx
11675                           <1>     ;push   ax
11676                           <1>     ; 12/04/2021
11677                           <1>     push    edx
11678                           <1>     push    eax
11679                           <1>     mov     ax, VBE_DISPI_INDEX_BANK
11680                           <1>     mov     dx, 1CEh ; VBE_DISPI_IOPORT_INDEX
11681                           <1>     out     dx, ax
11682                           <1>     ;pop    ax
11683                           <1>     ; 12/04/2021
11684                           <1>     pop     eax
11685                           <1>     ;;mov   dx, VBE_DISPI_IOPORT_DATA
11686                           <1>     ;mov    dl, 0CFh
11687                           <1>     inc     dl ; 1CFh = VBE_DISPI_IOPORT_DATA
11688                           <1>     out     dx, ax
11689                           <1>     in      ax, dx
11690                           <1>     ;pop    dx
11691                           <1>     ; 12/04/2021
11692                           <1>     pop     edx
11693                           <1>     cmp     dx, ax
11694                           <1>     jne     short dispi_set_bank_farcall_error
11695                           <1>     mov     ax, 004Fh
11696                           <1>     retn    ; retf for real mode far call
11697                           <1> dispi_set_bank_farcall_get:
11698                           <1>     mov     ax, VBE_DISPI_INDEX_BANK
11699                           <1>     ; 03/08/2022
11700                           <1>     mov     dx, 1CEh ; VBE_DISPI_IOPORT_INDEX
```

```
11701                                   <1>     out     dx, ax
11702                                   <1>     ;;mov   dx, VBE_DISPI_IOPORT_DATA
11703                                   <1>     ;mov    dl, 0CFh
11704                                   <1>     ; 03/08/2022
11705                                   <1>     inc     dl ; 1CFh = VBE_DISPI_IOPORT_DATA
11706                                   <1>     in      ax, dx
11707                                   <1>     mov     dx, ax
11708                                   <1>     retn    ; retf for real mode far call
11709                                   <1> dispi_set_bank_farcall_error:
11710                                   <1>     mov     ax, 014Fh
11711                                   <1>     retn    ; retf for real mode far call
11712                                   <1>
11713                                   <1> %endif
11714                                   <1>
11715                                   <1> ; % include 'vidata.s' ; VIDEO DATA
11716                                   <1>
11717                                   <1> ; /// End Of VIDEO FUNCTIONS ///
 2991
 2992                                       setup_rtc_int:
 2993                                       ; source: http://wiki.osdev.org/RTC
 2994 000040FC FA                          cli             ; disable interrupts
 2995                                       ; default int frequency is 1024 Hz (Lower 4 bits of register A is 0110b or 6)
 2996                                       ; in order to change this ...
 2997                                       ; frequency  = 32768 >> (rate-1) --> 32768 >> 5 = 1024
 2998                                       ; (rate must be above 2 and not over 15)
 2999                                       ; new rate = 15 --> 32768 >> (15-1) = 2 Hz
 3000 000040FD B08A                        mov     al, 8Ah
 3001 000040FF E670                        out     70h, al ; set index to register A, disable NMI
 3002 00004101 90                          nop
 3003 00004102 E471                        in      al, 71h ; get initial value of register A
 3004 00004104 88C4                        mov     ah, al
 3005 00004106 80E4F0                      and     ah, 0F0h
 3006 00004109 B08A                        mov     al, 8Ah
 3007 0000410B E670                        out     70h, al ; reset index to register A
 3008 0000410D 88E0                        mov     al, ah
 3009 0000410F 0C0F                        or      al, 0Fh ; new rate (0Fh -> 15)
 3010 00004111 E671                        out     71h, al ; write only our rate to A. Note, rate is the bottom 4 bits.
 3011                                       ; enable RTC interrupt
 3012 00004113 B08B                        mov     al, 8Bh ;
 3013 00004115 E670                        out     70h, al ; select register B and disable NMI
 3014 00004117 90                          nop
 3015 00004118 E471                        in      al, 71h ; read the current value of register B
 3016 0000411A 88C4                        mov     ah, al  ;
 3017 0000411C B08B                        mov     al, 8Bh ;
 3018 0000411E E670                        out     70h, al ; set the index again (a read will reset the index to register B)
 3019 00004120 88E0                        mov     al, ah  ;
 3020 00004122 0C40                        or      al, 40h ;
 3021 00004124 E671                        out     71h, al ; write the previous value ORed with 0x40. This turns on bit 6 of register B
 3022 00004126 FB                          sti
 3023 00004127 C3                          retn
 3024
 3025                                   ; Write memory information
 3026                                   ; 29/01/2016
 3027                                   ; 06/11/2014
 3028                                   ; 14/08/2015
 3029                                   memory_info:
 3030 00004128 A1[84760100]                mov     eax, [memory_size] ; in pages
 3031 0000412D 50                          push    eax
 3032 0000412E C1E00C                      shl     eax, 12           ; in bytes
 3033 00004131 BB0A000000                  mov     ebx, 10
 3034 00004136 89D9                        mov     ecx, ebx          ; 10
 3035 00004138 BE[41380100]                mov     esi, mem_total_b_str
 3036 0000413D E8D8000000                  call    bintdstr
 3037 00004142 58                          pop     eax
 3038 00004143 B107                        mov     cl, 7
 3039 00004145 BE[65380100]                mov     esi, mem_total_p_str
 3040 0000414A E8CB000000                  call    bintdstr
 3041                                       ; 14/08/2015
 3042 0000414F E8E3000000                  call    calc_free_mem
 3043                                       ; edx = calculated free pages
 3044                                       ; ecx = 0
 3045 00004154 A1[88760100]                mov     eax, [free_pages]
 3046 00004159 39D0                        cmp     eax, edx ; calculated free mem value
 3047                                               ; and initial free mem value are same or not?
 3048 0000415B 751D                        jne     short pmim ; print mem info with '?' if not
 3049 0000415D 52                          push    edx ; free memory in pages
 3050                                       ;mov    eax, edx
 3051 0000415E C1E00C                      shl     eax, 12 ; convert page count
 3052                                                       ; to byte count
 3053 00004161 B10A                        mov     cl, 10
 3054 00004163 BE[85380100]                mov     esi, free_mem_b_str
 3055 00004168 E8AD000000                  call    bintdstr
 3056 0000416D 58                          pop     eax
 3057 0000416E B107                        mov     cl, 7
 3058 00004170 BE[A9380100]                mov     esi, free_mem_p_str
 3059 00004175 E8A0000000                  call    bintdstr
 3060                                   pmim:
 3061 0000417A BE[2F380100]                mov     esi, msg_memory_info
 3062                                       ;
 3063 0000417F B407                        mov     ah, 07h ; Black background,
 3064                                                       ; light gray forecolor
 3065                                   print_kmsg: ; 29/01/2016
 3066 00004181 8825[AF760100]              mov     [ccolor], ah
 3067                                   pkmsg_loop:
 3068 00004187 AC                          lodsb
 3069 00004188 08C0                        or      al, al
 3070 0000418A 7410                        jz      short pkmsg_ok
 3071 0000418C 56                          push    esi
 3072                                       ; 13/05/2016
 3073 0000418D 0FB61D[AF760100]            movzx   ebx, byte [ccolor]
 3074                                                       ; Video page 0 (bh=0)
 3075 00004194 E842E1FFFF                  call    _write_tty
 3076 00004199 5E                          pop     esi
 3077 0000419A EBEB                        jmp     short pkmsg_loop
 3078                                   pkmsg_ok:
 3079 0000419C C3                          retn
 3080
 3081                                   ; 19/12/2020
 3082                                   ; temporary
 3083                                   ; Write default liner frame buffer address
 3084                                   ;
 3085                                   default_lfb_info:
 3086 0000419D 66A1[1A0F0000]              mov     ax, [def_LFB_addr] ; high word
 3087                                       ; 24/11/2023 - temporary
 3088 000041A3 6609C0                      or      ax, ax
 3089 000041A6 74F4                        jz      short pkmsg_ok
 3090 000041A8 E829000000                  call    wordtohex
 3091 000041AD A3[01390100]                mov     dword [lfb_addr_str], eax
 3092 000041B2 BE[EA380100]                mov     esi, msg_lfb_addr
 3093 000041B7 B40F                        mov     ah, 0Fh ; Black background,
 3094                                                       ; white forecolor
 3095 000041B9 EBC6                        jmp     short print_kmsg
 3096
 3097                                   ; Convert binary number to hexadecimal string
```

```
                                    ; 10/05/2015
                                    ; dsectpm.s (28/02/2015)
                                    ; Retro UNIX 386 v1 - Kernel v0.2.0.6
                                    ; 01/12/2014
                                    ; 25/11/2014
                                    ;
                                    bytetohex:
                                        ; INPUT ->
                                        ;       AL = byte (binary number)
                                        ; OUTPUT ->
                                        ;       AX = hexadecimal string
                                        ;
000041BB 53                             push    ebx
000041BC 31DB                           xor     ebx, ebx
000041BE 88C3                           mov     bl, al
000041C0 C0EB04                         shr     bl, 4
000041C3 8A9B[09420000]                 mov     bl, [ebx+hexchrs]
000041C9 86C3                           xchg    bl, al
000041CB 80E30F                         and     bl, 0Fh
000041CE 8AA3[09420000]                 mov     ah, [ebx+hexchrs]
000041D4 5B                             pop     ebx
000041D5 C3                             retn
                                    
                                    wordtohex:
                                        ; INPUT ->
                                        ;       AX = word (binary number)
                                        ; OUTPUT ->
                                        ;       EAX = hexadecimal string
                                        ;
000041D6 53                             push    ebx
000041D7 31DB                           xor     ebx, ebx
000041D9 86C4                           xchg    ah, al
000041DB 6650                           push    ax ; * save ax
000041DD 88E3                           mov     bl, ah
000041DF C0EB04                         shr     bl, 4
000041E2 8A83[09420000]                 mov     al, [ebx+hexchrs]
000041E8 88E3                           mov     bl, ah
000041EA 80E30F                         and     bl, 0Fh
000041ED 8AA3[09420000]                 mov     ah, [ebx+hexchrs]
000041F3 C1E010                         shl     eax, 16 ; ax -> hw of eax
000041F6 6658                           pop     ax ; * restore ax
000041F8 5B                             pop     ebx
000041F9 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
                                        ;
000041FB 50                             push    eax
000041FC C1E810                         shr     eax, 16
000041FF E8D2FFFFFF                     call    wordtohex
00004204 89C2                           mov     edx, eax
00004206 58                             pop     eax
                                        ;call   wordtohex
                                        ;retn
                                        ; 18/04/2021
00004207 EBCD                           jmp     short wordtohex
                                    
                                    ; 10/05/2015
                                    hex_digits:
                                    hexchrs:
00004209 303132333435363738-            db '0123456789ABCDEF'
00004212 39414243444546
                                    
                                    ; 19/01/2021 - VESA EDID ready flag (4Fh)
00004219 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)
0000421A 01CE                           add     esi, ecx
                                    btdstr0:
0000421C 4E                             dec     esi
0000421D 31D2                           xor     edx, edx
0000421F F7F3                           div     ebx
00004221 80C230                         add     dl, 30h
00004224 8816                           mov     [esi], dl
00004226 FEC9                           dec     cl
00004228 740C                           jz      short btdstr2 ; 08/09/2016
0000422A 09C0                           or      eax, eax
0000422C 75EE                           jnz     short btdstr0
                                    btdstr1:
0000422E 4E                             dec     esi
0000422F C60620                             mov     byte [esi], 20h ; blank space
00004232 FEC9                           dec     cl
00004234 75F8                           jnz     short btdstr1
                                    btdstr2:
00004236 C3                             retn
                                    
                                    ; Calculate free memory pages on M.A.T.
                                    ; 06/11/2014
                                    ; Temporary Code
                                    ;
                                    
                                    calc_free_mem:
00004237 31D2                           xor     edx, edx
                                        ;xor    ecx, ecx
00004239 668B0D[98760100]               mov     cx, [mat_size] ; in pages
00004240 C1E10A                         shl     ecx, 10 ; 1024 dwords per page
00004243 BE00001000                     mov     esi, MEM_ALLOC_TBL
                                    cfm0:
00004248 AD                             lodsd
00004249 51                             push    ecx
0000424A B920000000                     mov     ecx, 32
                                    cfm1:
0000424F D1E8                           shr     eax, 1
00004251 7301                           jnc     short cfm2
00004253 42                             inc     edx
```

```
3221                                    cfm2:
3222 00004254 E2F9                          loop    cfm1
3223 00004256 59                            pop     ecx
3224 00004257 E2EF                          loop    cfm0
3225 00004259 C3                            retn
3226
3227                                    %include 'diskio.s'  ; 07/03/2015
   1                                <1> ; ****************************************************************************
   2                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.8 - diskio.s
   3                                <1> ; ----------------------------------------------------------------------------
   4                                <1> ; Last Update: 23/06/2024 (Previous: 02/12/2023 - Kernel v2.0.7)
   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> ; diskio.inc (22/08/2015)
  15                                <1> ;
  16                                <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
  17                                <1> ; ****************************************************************************
  18                                <1> ; Ref: Retro UNIX 386 v1.1 (Kernel v0.2.1.5) 'diskio' modification: 12/07/2022
  19                                <1>
  20                                <1> ; Retro UNIX 386 v1 Kernel - DISKIO.INC
  21                                <1> ; Last Modification: 22/08/2015
  22                                <1> ;   (Initialized Disk Parameters Data is in 'DISKDATA.INC')
  23                                <1> ;   (Uninitialized Disk Parameters Data is in 'DISKBSS.INC')
  24                                <1>
  25                                <1> ; DISK I/O SYSTEM - Erdogan Tan (Retro UNIX 386 v1 project)
  26                                <1>
  27                                <1> ; ///////// DISK I/O SYSTEM ///////////////
  28                                <1>
  29                                <1> ; 11/04/2021
  30                                <1> ;; 06/02/2015
  31                                <1> ;diskette_io:
  32                                <1>     ;clc ; 20/07/2020
  33                                <1>     ;pushfd
  34                                <1>     ;push  cs
  35                                <1>     ;;call DISKETTE_IO_1
  36                                <1>     ;;retn
  37                                <1>
  38                                <1> ;;;;;; DISKETTE I/O ;;;;;;;;;;;;;;;;;;;; 06/02/2015 ;;;
  39                                <1> ;//////////////////////////////////////////////////////
  40                                <1>
  41                                <1> ; DISKETTE I/O - Erdogan Tan (Retro UNIX 386 v1 project)
  42                                <1> ; 20/02/2015
  43                                <1> ; 06/02/2015 (unix386.s)
  44                                <1> ; 16/12/2014 - 02/01/2015 (dsectrm2.s)
  45                                <1> ;
  46                                <1> ; Code (DELAY) modifications - AWARD BIOS 1999 (ADISK.EQU, COMMON.MAC)
  47                                <1> ;
  48                                <1> ; ADISK.EQU
  49                                <1>
  50                                <1> ;----- Wait control constants
  51                                <1>
  52                                <1> ;amount of time to wait while RESET is active.
  53                                <1>
  54                                <1> WAITCPU_RESET_ON   EQU     21              ;Reset on must last at least 14us
  55                                <1>                                            ;at 250 KBS xfer rate.
  56                                <1>                                            ;see INTEL MCS, 1985, pg. 5-456
  57                                <1>
  58                                <1> WAITCPU_FOR_STATUS EQU     100             ;allow 30 microseconds for
  59                                <1>                                            ;status register to become valid
  60                                <1>                                            ;before re-reading.
  61                                <1>
  62                                <1> ;After sending a byte to NEC, status register may remain
  63                                <1> ;incorrectly set for 24 us.
  64                                <1>
  65                                <1> WAITCPU_RQM_LOW            EQU     24              ;number of loops to check for
  66                                <1>                                            ;RQM low.
  67                                <1>
  68                                <1> ; COMMON.MAC
  69                                <1> ;
  70                                <1> ;   Timing macros
  71                                <1> ;
  72                                <1>
  73                                <1> %macro              SIODELAY 0                      ; SHORT IODELAY
  74                                <1>             jmp short $+2
  75                                <1> %endmacro
  76                                <1>
  77                                <1> %macro              IODELAY  0                      ; NORMAL IODELAY
  78                                <1>             jmp short $+2
  79                                <1>             jmp short $+2
  80                                <1> %endmacro
  81                                <1>
  82                                <1> %macro              NEWIODELAY 0
  83                                <1>             out 0EBh, al
  84                                <1> %endmacro
  85                                <1>
  86                                <1> ; (According to) AWARD BIOS 1999 - ATORGS.ASM (dw -> equ, db -> equ)
  87                                <1> ;;; WAIT_FOR_MEM
  88                                <1> ;WAIT_FDU_INT_LO   equ     017798          ; 2.5 secs in 30 micro units.
  89                                <1> ;WAIT_FDU_INT_HI   equ     1
  90                                <1> WAIT_FDU_INT_LH            equ     83334           ; 27/02/2015 (2.5 seconds waiting)
  91                                <1> ;;; WAIT_FOR_PORT
  92                                <1> ;WAIT_FDU_SEND_LO  equ     16667           ; .5 secons in 30 us units.
  93                                <1> ;WAIT_FDU_SEND_HI  equ     0
  94                                <1> WAIT_FDU_SEND_LH   equ     16667           ; 27/02/2015
  95                                <1> ;Time to wait while waiting for each byte of NEC results = .5
  96                                <1> ;seconds.  .5 seconds = 500,000 micros.  500,000/30 = 16,667.
  97                                <1> ;WAIT_FDU_RESULTS_LO       equ     16667           ; .5 seconds in 30 micro units.
  98                                <1> ;WAIT_FDU_RESULTS_HI       equ     0
  99                                <1> WAIT_FDU_RESULTS_LH        equ     16667  ; 27/02/2015
 100                                <1> ;;; WAIT_REFRESH
 101                                <1> ;amount of time to wait for head settle, per unit in parameter
 102                                <1> ;table = 1 ms.
 103                                <1> WAIT_FDU_HEAD_SETTLE       equ     33              ; 1 ms in 30 micro units.
 104                                <1>
 105                                <1>
 106                                <1> ; ///////////////// DISKETTE I/O ////////////////
 107                                <1>
 108                                <1> ; 11/12/2014 (copy from IBM PC-XT Model 286 BIOS - POSTEQU.INC)
 109                                <1>
 110                                <1> ;--------------------------------------
 111                                <1> ;   EQUATES USED BY POST AND BIOS :
 112                                <1> ;--------------------------------------
 113                                <1>
 114                                <1> ;--------- 8042 KEYBOARD INTERFACE AND DIAGNOSTIC CONTROL REGISTERS ------------
 115                                <1> ;PORT_A            EQU     060H            ; 8042 KEYBOARD SCAN CODE/CONTROL PORT
 116                                <1> ;PORT_B            EQU     061H            ; PORT B READ/WRITE DIAGNOSTIC REGISTER
 117                                <1> ;REFRESH_BIT       EQU     00010000B       ; REFRESH TEST BIT
```

```
118                     <1>
119                     <1> ;----------------------------------------
120                     <1> ;   CMOS EQUATES FOR THIS SYSTEM :
121                     <1> ;-------------------------------------------------------------------------------
122                     <1> ;CMOS_PORT  EQU     070H            ; I/O ADDRESS OF CMOS ADDRESS PORT
123                     <1> ;CMOS_DATA  EQU     071H            ; I/O ADDRESS OF CMOS DATA PORT
124                     <1> ;NMI        EQU     10000000B       ; DISABLE NMI INTERRUPTS MASK -
125                     <1>                                     ;  HIGH BIT OF CMOS LOCATION ADDRESS
126                     <1>
127                     <1> ;---------- CMOS TABLE LOCATION ADDRESS'S ## ---------------------------------
128                     <1> CMOS_DISKETTE       EQU     010H            ; DISKETTE DRIVE TYPE BYTE          ;
129                     <1> ;           EQU     011H            ; - RESERVED                            ;C
130                     <1> CMOS_DISK   EQU     012H            ; FIXED DISK TYPE BYTE                  ;H
131                     <1> ;           EQU     013H            ; - RESERVED                            ;E
132                     <1> CMOS_EQUIP  EQU     014H            ; EQUIPMENT WORD LOW BYTE               ;C
133                     <1>
134                     <1> ;---------- DISKETTE EQUATES -----------------------------------------------
135                     <1> INT_FLAG    EQU     10000000B       ; INTERRUPT OCCURRENCE FLAG
136                     <1> DSK_CHG     EQU     10000000B       ; DISKETTE CHANGE FLAG MASK BIT
137                     <1> DETERMINED  EQU     00010000B       ; SET STATE DETERMINED IN STATE BITS
138                     <1> HOME        EQU     00010000B       ; TRACK 0 MASK
139                     <1> SENSE_DRV_ST        EQU     00000100B       ; SENSE DRIVE STATUS COMMAND
140                     <1> TRK_SLAP    EQU     030H            ; CRASH STOP (48 TPI DRIVES)
141                     <1> QUIET_SEEK  EQU     00AH            ; SEEK TO TRACK 10
142                     <1> ;MAX_DRV    EQU     2               ; MAX NUMBER OF DRIVES
143                     <1> HD12_SETTLE EQU     15              ; 1.2 M HEAD SETTLE TIME
144                     <1> HD320_SETTLE        EQU     20              ; 320 K HEAD SETTLE TIME
145                     <1> MOTOR_WAIT  EQU     37              ; 2 SECONDS OF COUNTS FOR MOTOR TURN OFF
146                     <1>
147                     <1> ;---------- DISKETTE ERRORS -------------------------------------------------
148                     <1> ;TIME_OUT   EQU     080H            ; ATTACHMENT FAILED TO RESPOND
149                     <1> ;BAD_SEEK   EQU     040H            ; SEEK OPERATION FAILED
150                     <1> BAD_NEC     EQU     020H            ; DISKETTE CONTROLLER HAS FAILED
151                     <1> BAD_CRC     EQU     010H            ; BAD CRC ON DISKETTE READ
152                     <1> MED_NOT_FND EQU     00CH            ; MEDIA TYPE NOT FOUND
153                     <1> DMA_BOUNDARY        EQU     009H            ; ATTEMPT TO DMA ACROSS 64K BOUNDARY
154                     <1> BAD_DMA     EQU     008H            ; DMA OVERRUN ON OPERATION
155                     <1> MEDIA_CHANGE        EQU     006H            ; MEDIA REMOVED ON DUAL ATTACH CARD
156                     <1> RECORD_NOT_FND      EQU     004H            ; REQUESTED SECTOR NOT FOUND
157                     <1> WRITE_PROTECT       EQU     003H            ; WRITE ATTEMPTED ON WRITE PROTECT DISK
158                     <1> BAD_ADDR_MARK       EQU     002H            ; ADDRESS MARK NOT FOUND
159                     <1> BAD_CMD     EQU     001H            ; BAD COMMAND PASSED TO DISKETTE I/O
160                     <1>
161                     <1> ;---------- DISK CHANGE LINE EQUATES ----------------------------------------
162                     <1> NOCHGLN     EQU     001H            ; NO DISK CHANGE LINE AVAILABLE
163                     <1> CHGLN               EQU     002H            ; DISK CHANGE LINE AVAILABLE
164                     <1>
165                     <1> ;---------- MEDIA/DRIVE STATE INDICATORS -------------------------------------
166                     <1> TRK_CAPA    EQU     00000001B       ; 80 TRACK CAPABILITY
167                     <1> FMT_CAPA    EQU     00000010B       ; MULTIPLE FORMAT CAPABILITY (1.2M)
168                     <1> DRV_DET     EQU     00000100B       ; DRIVE DETERMINED
169                     <1> MED_DET     EQU     00010000B       ; MEDIA DETERMINED BIT
170                     <1> DBL_STEP    EQU     00100000B       ; DOUBLE STEP BIT
171                     <1> RATE_MSK    EQU     11000000B       ; MASK FOR CLEARING ALL BUT RATE
172                     <1> RATE_500    EQU     00000000B       ; 500 KBS DATA RATE
173                     <1> RATE_300    EQU     01000000B       ; 300 KBS DATA RATE
174                     <1> RATE_250    EQU     10000000B       ; 250 KBS DATA RATE
175                     <1> STRT_MSK    EQU     00001100B       ; OPERATION START RATE MASK
176                     <1> SEND_MSK    EQU     11000000B       ; MASK FOR SEND RATE BITS
177                     <1>
178                     <1> ;---------- MEDIA/DRIVE STATE INDICATORS COMPATIBILITY ------------------------
179                     <1> M3D3U               EQU     00000000B       ; 360 MEDIA/DRIVE NOT ESTABLISHED
180                     <1> M3D1U               EQU     00000001B       ; 360 MEDIA,1.2DRIVE NOT ESTABLISHED
181                     <1> M1D1U               EQU     00000010B       ; 1.2 MEDIA/DRIVE NOT ESTABLISHED
182                     <1> MED_UNK     EQU     00000111B       ; NONE OF THE ABOVE
183                     <1>
184                     <1> ;---------- INTERRUPT EQUATES -----------------------------------------------
185                     <1> ;EOI        EQU     020H            ; END OF INTERRUPT COMMAND TO 8259
186                     <1> ;INTA00             EQU     020H            ; 8259 PORT
187                     <1> INTA01              EQU     021H            ; 8259 PORT
188                     <1> INTB00              EQU     0A0H            ; 2ND 8259
189                     <1> INTB01              EQU     0A1H            ;
190                     <1>
191                     <1> ;-------------------------------------------------------------------------------
192                     <1> DMA08               EQU     008H            ; DMA STATUS REGISTER PORT ADDRESS
193                     <1> DMA         EQU     000H            ; DMA CH.0 ADDRESS REGISTER PORT ADDRESS
194                     <1> DMA18               EQU     0D0H            ; 2ND DMA STATUS PORT ADDRESS
195                     <1> DMA1        EQU     0C0H            ; 2ND DMA CH.0 ADDRESS REGISTER ADDRESS
196                     <1> ;-------------------------------------------------------------------------------
197                     <1> ;TIMER              EQU     040H            ; 8254 TIMER - BASE ADDRESS
198                     <1>
199                     <1> ;-------------------------------------------------------------------------------
200                     <1> DMA_PAGE    EQU     081H            ; START OF DMA PAGE REGISTERS
201                     <1>
202                     <1> ; 06/02/2015 (unix386.s, protected mode modifications)
203                     <1> ; (unix386.s <-- dsectrm2.s)
204                     <1> ; 11/12/2014 (copy from IBM PC-XT Model 286 BIOS - DSEG.INC)
205                     <1>
206                     <1> ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
207                     <1> ; 10/12/2014
208                     <1> ;
209                     <1> ;int40h:
210                     <1> ;   pushf
211                     <1> ;   push    cs
212                     <1> ;   ;cli
213                     <1> ;   call    DISKETTE_IO_1
214                     <1> ;   retn
215                     <1>
216                     <1> ; DSKETTE ----- 04/21/86 DISKETTE BIOS
217                     <1> ; (IBM PC XT Model 286 System BIOS Source Code, 04-21-86)
218                     <1> ;
219                     <1>
220                     <1> ;-- INT13H -------------------------------------------------------------------
221                     <1> ; DISKETTE I/O
222                     <1> ;   THIS INTERFACE PROVIDES ACCESS TO THE 5 1/4 INCH 360 KB,
223                     <1> ;   1.2 MB, 720 KB AND 1.44 MB DISKETTE DRIVES.
224                     <1> ; INPUT
225                     <1> ;   (AH) =  00H RESET DISKETTE SYSTEM
226                     <1> ;           HARD RESET TO NEC, PREPARE COMMAND, RECALIBRATE REQUIRED
227                     <1> ;           ON ALL DRIVES
228                     <1> ;-------------------------------------------------------------------------------
229                     <1> ;   (AH)= 01H  READ THE STATUS OF THE SYSTEM INTO (AH)
230                     <1> ;           @DISKETTE_STATUS FROM LAST OPERATION IS USED
231                     <1> ;-------------------------------------------------------------------------------
232                     <1> ;   REGISTERS FOR READ/WRITE/VERIFY/FORMAT
233                     <1> ;   (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
234                     <1> ;   (DH) - HEAD NUMBER (0-1 ALLOWED, NOT VALUE CHECKED)
235                     <1> ;   (CH) - TRACK NUMBER (NOT VALUE CHECKED)
236                     <1> ;           MEDIA   DRIVE   TRACK NUMBER
237                     <1> ;           320/360 320/360     0-39
238                     <1> ;           320/360 1.2M        0-39
239                     <1> ;           1.2M    1.2M        0-79
240                     <1> ;           720K    720K        0-79
241                     <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
<1> ;       (AX) - 0
<1> ;    UNDER THE FOLLOWING CIRCUMSTANCES:
<1> ;       (1) THE DRIVE NUMBER IS INVALID,
<1> ;       (2) THE DRIVE TYPE IS UNKNOWN AND CMOS IS NOT PRESENT,
<1> ;       (3) THE DRIVE TYPE IS UNKNOWN AND CMOS IS BAD,
<1> ;       (4) OR THE DRIVE TYPE IS UNKNOWN AND THE CMOS DRIVE TYPE IS INVALID
<1> ;       THEN ES,AX,BX,CX,DH,DI=0 ; DL=NUMBER OF DRIVES.
<1> ;       IF NO DRIVES ARE PRESENT THEN: ES,AX,BX,CX,DX,DI=0.
<1> ;       @DISKETTE_STATUS = 0 AND CY IS RESET.
<1> ;--------------------------------------------------------------------------
<1> ;   (AH)= 15H  READ DASD TYPE
<1> ;   OUTPUT REGISTERS
<1> ;   (AH) - ON RETURN IF CARRY FLAG NOT SET, OTHERWISE ERROR
<1> ;          00 - DRIVE NOT PRESENT
<1> ;          01 - DISKETTE, NO CHANGE LINE AVAILABLE
<1> ;          02 - DISKETTE, CHANGE LINE AVAILABLE
<1> ;          03 - RESERVED (FIXED DISK)
<1> ;   (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
<1> ;--------------------------------------------------------------------------
<1> ;   (AH)= 16H  DISK CHANGE LINE STATUS
<1> ;   OUTPUT REGISTERS
<1> ;   (AH) - 00 - DISK CHANGE LINE NOT ACTIVE
<1> ;          06 - DISK CHANGE LINE ACTIVE & CARRY BIT ON
<1> ;   (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
<1> ;--------------------------------------------------------------------------
<1> ;   (AH)= 17H  SET DASD TYPE FOR FORMAT
<1> ;   INPUT REGISTERS
<1> ;   (AL) -  00 - NOT USED
<1> ;          01 - DISKETTE 320/360K IN 360K DRIVE
<1> ;          02 - DISKETTE 360K IN 1.2M DRIVE
<1> ;          03 - DISKETTE 1.2M IN 1.2M DRIVE
<1> ;          04 - DISKETTE 720K IN 720K DRIVE
<1> ;   (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED:
<1> ;          (DO NOT USE WHEN DISKETTE ATTACH CARD USED)
<1> ;--------------------------------------------------------------------------
<1> ;   (AH)= 18H  SET MEDIA TYPE FOR FORMAT
<1> ;   INPUT REGISTERS
<1> ;   (CH) - LOW ORDER 8 OF 10 BITS MAXIMUM TRACKS
<1> ;   (CL) - BITS 7 & 6 - HIGH ORDER TWO BITS OF MAXIMUM TRACKS
<1> ;          BITS 5 THRU 0 - MAXIMUM SECTORS PER TRACK
<1> ;   (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHACKED)
<1> ;   OUTPUT REGISTERS:
<1> ;   (ES:DI) - POINTER TO DRIVE PARAMETERS TABLE FOR THIS MEDIA TYPE,
<1> ;             UNCHANGED IF (AH) IS NON-ZERO
<1> ;   (AH) - 00H, CY = 0, TRACK AND SECTORS/TRACK COMBINATION IS SUPPORTED
<1> ;        - 01H, CY = 1, FUNCTION IS NOT AVAILABLE
<1> ;        - 0CH, CY = 1, TRACK AND SECTORS/TRACK COMBINATION IS NOT SUPPORTED
```

```
366                                     <1> ;       - 80H, CY = 1, TIME OUT (DISKETTE NOT PRESENT)
367                                     <1> ;-------------------------------------------------------------------------------
368                                     <1> ;   DISK CHANGE STATUS IS ONLY CHECKED WHEN A MEDIA SPECIFIED IS OTHER
369                                     <1> ;   THAN 360 KB DRIVE. IF THE DISK CHANGE LINE IS FOUND TO BE
370                                     <1> ;   ACTIVE THE FOLLOWING ACTIONS TAKE PLACE:
371                                     <1> ;           ATTEMPT TO RESET DISK CHANGE LINE TO INACTIVE STATE.
372                                     <1> ;           IF ATTEMPT SUCCEEDS SET DASD TYPE FOR FORMAT AND RETURN DISK
373                                     <1> ;           CHANGE ERROR CODE
374                                     <1> ;           IF ATTEMPT FAILS RETURN TIMEOUT ERROR CODE AND SET DASD TYPE
375                                     <1> ;           TO A PREDETERMINED STATE INDICATING MEDIA TYPE UNKNOWN.
376                                     <1> ;   IF THE DISK CHANGE LINE IN INACTIVE PERFORM SET DASD TYPE FOR FORMAT.
377                                     <1> ;
378                                     <1> ; DATA VARIABLE -- @DISK_POINTER
379                                     <1> ;   DOUBLE WORD POINTER TO THE CURRENT SET OF DISKETTE PARAMETERS
380                                     <1> ;-------------------------------------------------------------------------------
381                                     <1> ; OUTPUT FOR ALL FUNCTIONS
382                                     <1> ;   AH = STATUS OF OPERATION
383                                     <1> ;           STATUS BITS ARE DEFINED IN THE EQUATES FOR @DISKETTE_STATUS
384                                     <1> ;           VARIABLE IN THE DATA SEGMENT OF THIS MODULE
385                                     <1> ;   CY = 0  SUCCESSFUL OPERATION (AH=0 ON RETURN, EXCEPT FOR READ DASD
386                                     <1> ;           TYPE AH=(15)).
387                                     <1> ;   CY = 1  FAILED OPERATION (AH HAS ERROR REASON)
388                                     <1> ;   FOR READ/WRITE/VERIFY
389                                     <1> ;           DS,BX,DX,CX PRESERVED
390                                     <1> ;   NOTE: IF AN ERROR IS REPORTED BY THE DISKETTE CODE, THE APPROPRIATE
391                                     <1> ;           ACTION IS TO RESET THE DISKETTE, THEN RETRY THE OPERATION.
392                                     <1> ;           ON READ ACCESSES, NO MOTOR START DELAY IS TAKEN, SO THAT
393                                     <1> ;           THREE RETRIES ARE REQUIRED ON READS TO ENSURE THAT THE
394                                     <1> ;           PROBLEM IS NOT DUE TO MOTOR START-UP.
395                                     <1> ;-------------------------------------------------------------------------------
396                                     <1> ;
397                                     <1> ; DISKETTE STATE MACHINE - ABSOLUTE ADDRESS 40:90 (DRIVE A) & 91 (DRIVE B)
398                                     <1> ;
399                                     <1> ;   ---------------------------------------------------------------
400                                     <1> ;   |       |       |       |       |       |       |       |       |
401                                     <1> ;   |   7   |   6   |   5   |   4   |   3   |   2   |   1   |   0   |
402                                     <1> ;   |       |       |       |       |       |       |       |       |
403                                     <1> ;   ---------------------------------------------------------------
404                                     <1> ;   |       |       |       |       |       |       |       |       |
405                                     <1> ;   |       |       |       |       |       ---------------------
406                                     <1> ;   |       |       |       |       |               |
407                                     <1> ;   |       |       |       |   RESERVED            |
408                                     <1> ;   |       |       |       |                   PRESENT STATE
409                                     <1> ;   |       |       |       |               000: 360K IN 360K DRIVE UNESTABLISHED
410                                     <1> ;   |       |       |       |               001: 360K IN 1.2M DRIVE UNESTABLISHED
411                                     <1> ;   |       |       |       |               010: 1.2M IN 1.2M DRIVE UNESTABLISHED
412                                     <1> ;   |       |       |       |               011: 360K IN 360K DRIVE ESTABLISHED
413                                     <1> ;   |       |       |       |               100: 360K IN 1.2M DRIVE ESTABLISHED
414                                     <1> ;   |       |       |       |               101: 1.2M IN 1.2M DRIVE ESTABLISHED
415                                     <1> ;   |       |       |       |               110: RESERVED
416                                     <1> ;   |       |       |       |               111: NONE OF THE ABOVE
417                                     <1> ;   |       |       |       |
418                                     <1> ;   |       |       |       ------>MEDIA/DRIVE ESTABLISHED
419                                     <1> ;   |       |       |
420                                     <1> ;   |       |       --------------->       DOUBLE STEPPING REQUIRED (360K IN 1.2M
421                                     <1> ;   |       |                       DRIVE)
422                                     <1> ;   |       |
423                                     <1> ;   -------------------------------->      DATA TRANSFER RATE FOR THIS DRIVE:
424                                     <1> ;
425                                     <1> ;                                         00: 500 KBS
426                                     <1> ;                                         01: 300 KBS
427                                     <1> ;                                         10: 250 KBS
428                                     <1> ;                                         11: RESERVED
429                                     <1> ;
430                                     <1> ;
431                                     <1> ;-------------------------------------------------------------------------------
432                                     <1> ; STATE OPERATION STARTED - ABSOLUTE ADDRESS 40:92 (DRIVE A) & 93 (DRIVE B)
433                                     <1> ;-------------------------------------------------------------------------------
434                                     <1> ; PRESENT CYLINDER NUMBER - ABSOLUTE ADDRESS 40:94 (DRIVE A) & 95 (DRIVE B)
435                                     <1> ;-------------------------------------------------------------------------------
436                                     <1>
437                                     <1> struc MD
438 00000000 ??                         <1>     .SPEC1          resb    1       ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
439 00000001 ??                         <1>     .SPEC2          resb    1       ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
440 00000002 ??                         <1>     .OFF_TIM        resb    1       ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
441 00000003 ??                         <1>     .BYT_SEC        resb    1       ; 512 BYTES/SECTOR
442 00000004 ??                         <1>     .SEC_TRK        resb    1       ; EOT (LAST SECTOR ON TRACK)
443 00000005 ??                         <1>     .GAP            resb    1       ; GAP LENGTH
444 00000006 ??                         <1>     .DTL            resb    1       ; DTL
445 00000007 ??                         <1>     .GAP3           resb    1       ; GAP LENGTH FOR FORMAT
446 00000008 ??                         <1>     .FIL_BYT        resb    1       ; FILL BYTE FOR FORMAT
447 00000009 ??                         <1>     .HD_TIM         resb    1       ; HEAD SETTLE TIME (MILLISECONDS)
448 0000000A ??                         <1>     .STR_TIM        resb    1       ; MOTOR START TIME (1/8 SECONDS)
449 0000000B ??                         <1>     .MAX_TRK        resb    1       ; MAX. TRACK NUMBER
450 0000000C ??                         <1>     .RATE           resb    1       ; DATA TRANSFER RATE
451                                     <1> endstruc
452                                     <1>
453                                     <1> BIT7OFF     EQU     7FH
454                                     <1> BIT7ON      EQU     80H
455                                     <1>
456                                     <1> ; 30/08/2020 - TRDOS 386 v2
457                                     <1>
458                                     <1> ;;int13h: ; 16/02/2015
459                                     <1> ;; 16/02/2015 - 21/02/2015
460                                     <1> ; 17/07/2022 - TRDOS 386 v2.0.5
461                                     <1> ;int40h:
462                                     <1> ;; 11/04/2021
463                                     <1> ;diskette_io:
464                                     <1> ;   clc ; 20/07/2020
465                                     <1> ;   pushfd
466                                     <1> ;   push    cs
467                                     <1> ;   call    DISKETTE_IO_1
468                                     <1> ;   retn
469                                     <1>
470                                     <1>     ; 09/08/2022
471                                     <1>     ; 06/08/2022
472                                     <1>     ; 17/07/2022 - TRDOS 386 v2.0.5
473                                     <1>     ; (jump from DISK_IO)
474                                     <1> DISKETTE_IO_1:
475                                     <1>     ;sti ; 17/07/2022              ; INTERRUPTS BACK ON
476                                     <1>
477                                     <1>     ;push   ebp                     ; USER REGISTER
478                                     <1>     ;push   edi                     ; USER REGISTER
479                                     <1>     ;push   edx                     ; HEAD #, DRIVE # OR USER REGISTER
480                                     <1>     ;push   ebx                     ; BUFFER OFFSET PARAMETER OR REGISTER
481                                     <1>     ;push   ecx                     ; TRACK #-SECTOR # OR USER REGISTER
482                                     <1>     ;mov    ebp, esp               ; BP     => PARAMETER LIST DEP. ON AH
483                                     <1>                                     ; [BP]   = SECTOR #
484                                     <1>                                     ; [BP+1] = TRACK #
485                                     <1>                                     ; [BP+2] = BUFFER OFFSET
486                                     <1>                                     ; FOR RETURN OF DRIVE PARAMETERS:
487                                     <1>                                     ; CL/[BP] = BITS 7&6 HI BITS OF MAX CYL
488                                     <1>                                     ;           BITS 0-5 MAX SECTORS/TRACK
489                                     <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
                        <1>     ;push   es ; 06/02/2015
                        <1>     ;push   ds                      ; BUFFER SEGMENT PARM OR USER REGISTER
                        <1>     ;push   esi                     ; USER REGISTERS
                        <1>     ;;call DDS                      ; SEGMENT OF BIOS DATA AREA TO DS
                        <1>     ;;mov   cx, cs
                        <1>     ;;mov   ds, cx
                        <1>     ;mov    cx, KDATA
                        <1>     ;mov    ds, cx
                        <1>     ;mov    es, cx
                        <1>
                        <1>     ; 17/07/2022
                        <1>     ; Registers are also on stack
                        <1>     ; (with same contents)
                        <1>     ; in following order:
                        <1>     ;
                        <1>     ;    ebx = esp+20
                        <1>     ;    ecx = esp+16
                        <1>     ;    edx = esp+12
                        <1>     ;    esi = esp+8
                        <1>     ;    edi = esp+4
                        <1>     ;
                        <1>     ; [esp] = caller's return address (from 'DISK_IO')
                        <1>     ;
                        <1>     ; cs = KCODE == KDATA
                        <1>     ; ds = es = ss = KDATA
                        <1>
                        <1>     ; 17/07/2022
0000425A 55             <1>     push    ebp
0000425B 89DD           <1>     mov     ebp, ebx
                        <1>
                        <1>     ;cmp    ah, (FNC_TAE-FNC_TAB)/2       ; CHECK FOR > LARGEST FUNCTION
0000425D 80FC19         <1>     cmp     ah, (FNC_TAE-FNC_TAB)/4       ; 18/02/2015
                        <1>     ;jb     short OK_FUNC           ; FUNCTION OK
                        <1>     ;mov    ah, 14h                 ; REPLACE WITH KNOWN INVALID FUNCTION
                        <1>     ; 09/08/2022
00004260 730F           <1>     jnb     short INV_FUNC
                        <1> OK_FUNC:
00004262 80FC01         <1>     cmp     ah, 1                   ; RESET OR STATUS ?
00004265 760C           <1>     jbe     short OK_DRV            ; IF RESET OR STATUS DRIVE ALWAYS OK
00004267 80FC08         <1>     cmp     ah, 8                   ; READ DRIVE PARMS ?
0000426A 7407           <1>     je      short OK_DRV            ; IF SO DRIVE CHECKED LATER
0000426C 80FA01         <1>     cmp     dl, 1                   ; DRIVES 0 AND 1 OK
0000426F 7602           <1>     jbe     short OK_DRV            ; IF 0 OR 1 THEN JUMP
                        <1> INV_FUNC:
00004271 B414           <1>     mov     ah, 14h                 ; REPLACE WITH KNOWN INVALID FUNCTION
                        <1> OK_DRV:
                        <1>     ; 17/07/2022
                        <1>     ;xor    ecx, ecx
                        <1>     ;;mov   esi, ecx ; 08/02/2015
                        <1>     ;mov    edi, ecx ; 08/02/2015
                        <1>     ;mov    cl, ah                  ; CL = FUNCTION
                        <1>     ;;xor   ch, ch                  ; CX = FUNCTION
                        <1>     ;;shl   cl, 1                   ; FUNCTION TIMES 2
                        <1>     ;shl    cl, 2 ; 20/02/2015      ; FUNCTION TIMES 4 (for 32 bit offset)
                        <1>     ;mov    ebx, FNC_TAB            ; LOAD START OF FUNCTION TABLE
                        <1>     ;add    ebx, ecx                ; ADD OFFSET INTO TABLE => ROUTINE
                        <1>
                        <1>     ; 17/07/2022
00004273 29DB           <1>     sub     ebx, ebx
00004275 88E3           <1>     mov     bl, ah                  ; BL = FUNCTION
00004277 C0E302         <1>     shl     bl, 2 ; * 4
0000427A 81C3[99420000] <1>     add     ebx, FNC_TAB            ; [EBX] = FUNCTION ADDRESS
                        <1>
00004280 88F4           <1>     mov     ah, dh                  ; AX = HEAD #,# OF SECTORS OR DASD TYPE
                        <1>     ;xor    dh, dh                  ; DX = DRIVE #
                        <1>     ;mov    si, ax                  ; SI = HEAD #,# OF SECTORS OR DASD TYPE
                        <1>     ;mov    di, dx                  ; DI = DRIVE #
                        <1>
00004282 0FB7F0         <1>     movzx   esi, ax                 ; ESI = HEAD #,# OF SECTORS OR DASD TYPE
00004285 0FB6FA         <1>     movzx   edi, dl                 ; EDI = DRIVE #
                        <1>
                        <1>     ; CH = cylinder number (low 8 bit)
                        <1>     ; CL = sector number (and high 2 bits of cylinder number)
                        <1>
                        <1>     ; 06/08/2022
                        <1>     ; 11/12/2014
                        <1>       ;mov       [cfd], dl               ; current floppy drive (for 'GET_PARM')
                        <1>     ; 06/08/2022
                        <1>     ; EDI = (current) DRIVE #
                        <1>     ;
00004288 8A25[08770100] <1>     mov     ah, [DSKETTE_STATUS]    ; LOAD STATUS TO AH FOR STATUS FUNCTION
0000428E C605[08770100]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>     ; 17/07/2022
                        <1>     ; EBX = pointer to function address
                        <1>     ; EBP = buffer address
                        <1>     ; EDI = drive number (0 or 1)
                        <1>     ; ESI = head number (byte 1) and sector count or disk type (byte 0)
                        <1>     ; CH = cylinder number (low 8 bit)
                        <1>     ; CL = sector number (and high 2 bits of cylinder number)
                        <1>
                        <1>                                     ; (AH) = @DSKETTE_STATUS
00004295 FF13           <1>     call    dword [ebx]             ; CALL THE REQUESTED FUNCTION
                        <1>
                        <1>     ;pop    esi                     ; RESTORE ALL REGISTERS
                        <1>     ;pop    ds
                        <1>     ;pop    es      ; 06/02/2015
                        <1>     ;pop    ecx
                        <1>     ;pop    ebx
                        <1>     ;pop    edx
                        <1>     ;pop    edi
```

```
                                <1>     ;mov    ebp, esp
                                <1>     ;push   eax
                                <1>     ;pushfd
                                <1>     ;pop    eax
                                <1>     ;;mov   [bp+6], ax
                                <1>     ;mov    [ebp+12], eax  ; 18/02/2015, flags
                                <1>     ;pop    eax
                                <1>     ;pop    ebp
                                <1>     ;iretd
                                <1>
                                <1>     ; 17/07/2022
00004297 5D                     <1>     pop     ebp
                                <1>
                                <1>     ; 17/07/2022
                                <1>     ; Stack order:
                                <1>     ;    ebx = esp+20
                                <1>     ;    ecx = esp+16
                                <1>     ;    edx = esp+12
                                <1>     ;    esi = esp+8
                                <1>     ;    edi = esp+4
                                <1>     ;
                                <1>     ; [esp] = caller's return address (from 'DISK_IO')
                                <1>
                                <1>     ; CF = disk i/o status (1 = error)
                                <1>     ; AH = error code (if > 0 and cf = 1)
                                <1>
                                <1>     ; 17/07/2022
00004298 C3                     <1>     retn    ; return to the caller of 'DISK_IO'
                                <1>
                                <1> ;--------------------------------------------------------------------------------
                                <1> ; DW --> dd (06/02/2015)
00004299 [FD420000]             <1> FNC_TAB     dd      DSK_RESET               ; AH = 00H; RESET
0000429D [68430000]             <1>     dd      DSK_STATUS              ; AH = 01H; STATUS
000042A1 [78430000]             <1>     dd      DSK_READ                ; AH = 02H; READ
000042A5 [85430000]             <1>     dd      DSK_WRITE               ; AH = 03H; WRITE
000042A9 [99440000]             <1>     dd      DSK_VERF                ; AH = 04H; VERIFY
000042AD [A9440000]             <1>     dd      DSK_FORMAT              ; AH = 05H; FORMAT
000042B1 [27450000]             <1>     dd      FNC_ERR                 ; AH = 06H; INVALID
000042B5 [27450000]             <1>     dd      FNC_ERR                 ; AH = 07H; INVALID
000042B9 [33450000]             <1>     dd      DSK_PARMS               ; AH = 08H; READ DRIVE PARAMETERS
000042BD [27450000]             <1>     dd      FNC_ERR                 ; AH = 09H; INVALID
000042C1 [27450000]             <1>     dd      FNC_ERR                 ; AH = 0AH; INVALID
000042C5 [27450000]             <1>     dd      FNC_ERR                 ; AH = 0BH; INVALID
000042C9 [27450000]             <1>     dd      FNC_ERR                 ; AH = 0CH; INVALID
000042CD [27450000]             <1>     dd      FNC_ERR                 ; AH = 0DH; INVALID
000042D1 [27450000]             <1>     dd      FNC_ERR                 ; AH = 0EH; INVALID
000042D5 [27450000]             <1>     dd      FNC_ERR                 ; AH = 0FH; INVALID
000042D9 [27450000]             <1>     dd      FNC_ERR                 ; AH = 10H; INVALID
000042DD [27450000]             <1>     dd      FNC_ERR                 ; AH = 11H; INVALID
000042E1 [27450000]             <1>     dd      FNC_ERR                 ; AH = 12H; INVALID
000042E5 [27450000]             <1>     dd      FNC_ERR                 ; AH = 13H; INVALID
000042E9 [27450000]             <1>     dd      FNC_ERR                 ; AH = 14H; INVALID
000042ED [F0450000]             <1>     dd      DSK_TYPE                ; AH = 15H; READ DASD TYPE
000042F1 [10460000]             <1>     dd      DSK_CHANGE              ; AH = 16H; CHANGE STATUS
000042F5 [3F460000]             <1>     dd      FORMAT_SET              ; AH = 17H; SET DASD TYPE
000042F9 [B4460000]             <1>     dd      SET_MEDIA               ; AH = 18H; SET MEDIA TYPE
                                <1> FNC_TAE EQU     $                       ; END
                                <1>
                                <1>     ; 17/07/2022 - TRDOS 386 v2.0.5
                                <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:
                                <1>     ; 17/07/2022
000042FD 66BAF203               <1>     mov     dx, 03F2h               ; ADAPTER CONTROL PORT
00004301 FA                     <1>     cli                             ; NO INTERRUPTS
00004302 A0[06770100]           <1>     mov     al, [MOTOR_STATUS]      ; GET DIGITAL OUTPUT REGISTER REFLECTION
00004307 243F                   <1>     and     al, 00111111b           ; KEEP SELECTED AND MOTOR ON BITS
00004309 C0C004                 <1>     rol     al, 4                   ; MOTOR VALUE TO HIGH NIBBLE
                                <1>                                     ; DRIVE SELECT TO LOW NIBBLE
0000430C 0C08                   <1>     or      al, 00001000b           ; TURN ON INTERRUPT ENABLE
0000430E EE                     <1>     out     dx, al                  ; RESET THE ADAPTER
0000430F C605[05770100]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
 83 00004316 E6EB               <2>  out 0EBh, al
                                <1>
                                <1>     ; 17/12/2014
                                <1>     ; AWARD BIOS 1999 - RESETDRIVES (ADISK.ASM)
00004318 B915000000             <1>     mov     ecx, WAITCPU_RESET_ON ; cx = 21 -- Min. 14 micro seconds !?
                                <1> wdw1:
                                <1>     NEWIODELAY   ; 27/02/2015
 83 0000431D E6EB               <2>  out 0EBh, al
0000431F E2FC                   <1>     loop    wdw1
                                <1>     ;
00004321 0C04                   <1>     or      al, 00000100b           ; TURN OFF RESET BIT
00004323 EE                     <1>     out     dx, al                  ; RESET THE ADAPTER
                                <1>     ; 16/12/2014
                                <1>     IODELAY
 78 00004324 EB00               <2>  jmp short $+2
 79 00004326 EB00               <2>  jmp short $+2
                                <1>     ;
                                <1>     ;sti                            ; ENABLE THE INTERRUPTS
00004328 E8DD0A0000             <1>     call    WAIT_INT                ; WAIT FOR THE INTERRUPT
0000432D 7230                   <1>     jc      short DR_ERR            ; IF ERROR, RETURN IT
                                <1>     ;mov    cx, 11000000b           ; CL = EXPECTED @NEC_STATUS
                                <1>     ; 17/07/2022
                                <1>     ;xor    ch, ch
0000432F B1C0                   <1>     mov     cl, 11000000b
                                <1> NXT_DRV:
                                <1>     ;push   cx                      ; SAVE FOR CALL
                                <1>     ; 11/04/2021
00004331 51                     <1>     push    ecx
00004332 B8[5E430000]           <1>     mov     eax, DR_POP_ERR         ; LOAD NEC_OUTPUT ERROR ADDRESS
00004337 50                     <1>     push    eax                     ; "
00004338 B408                   <1>     mov     ah, 08h                 ; SENSE INTERRUPT STATUS COMMAND
0000433A E8C3090000             <1>     call    NEC_OUTPUT
0000433F 58                     <1>     pop     eax                     ; THROW AWAY ERROR RETURN
00004340 E8F40A0000             <1>     call    RESULTS                 ; READ IN THE RESULTS
                                <1>     ;pop    cx                      ; RESTORE AFTER CALL
                                <1>     ; 11/04/2021
00004345 59                     <1>     pop     ecx
00004346 7217                   <1>     jc      short DR_ERR            ; ERROR RETURN
00004348 3A0D[09770100]         <1>     cmp     cl, [NEC_STATUS]        ; TEST FOR DRIVE READY TRANSITION
0000434E 750F                   <1>     jnz     short DR_ERR            ; EVERYTHING OK
00004350 FEC1                   <1>     inc     cl                      ; NEXT EXPECTED @NEC_STATUS
00004352 80F9C3                 <1>     cmp     cl, 11000011b           ; ALL POSSIBLE DRIVES CLEARED
00004355 76DA                   <1>     jbe     short NXT_DRV           ; FALL THRU IF 11000100B OR >
```

```
                                <1>     ;
00004357 E800040000             <1>     call    SEND_SPEC               ; SEND SPECIFY COMMAND TO NEC
                                <1> RESBAC:
                                <1>     ; 06/08/2022
0000435C EB10                   <1>     jmp     short SETUP_END_X
                                <1>     ;call   SETUP_END               ; VARIOUS CLEANUPS
                                <1>     ;;mov   bx, si                  ; GET SAVED AL TO BL
                                <1>     ;; 17/07/2022
                                <1>     ;mov    ebx, esi
                                <1>     ;mov    al, bl                  ; PUT BACK FOR RETURN
                                <1>     ;retn
                                <1>
                                <1> DR_POP_ERR:
                                <1>     ;pop    cx                      ; CLEAR STACK
                                <1>     ; 11/04/2021
0000435E 59                     <1>     pop     ecx
                                <1> DR_ERR:
0000435F 800D[08770100]20       <1>     or      byte [DSKETTE_STATUS], BAD_NEC ; SET ERROR CODE
                                <1>     ;jmp    short RESBAC            ; RETURN FROM RESET
                                <1>     ; 06/08/2022
00004366 EB06                   <1>     jmp     short SETUP_END_X
                                <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:
00004368 8825[08770100]         <1>     mov     [DSKETTE_STATUS], ah  ; PUT BACK FOR SETUP END
                                <1> SETUP_END_X:        ; 06/08/2022
0000436E E809010000             <1>     call    SETUP_END               ; VARIOUS CLEANUPS
                                <1>     ;mov    bx, si                  ; GET SAVED AL TO BL
                                <1>     ;mov    al, bl                  ; PUT BACK FOR RETURN
                                <1>     ; 06/08/2022
00004373 89F3                   <1>     mov     ebx, esi
00004375 88D8                   <1>     mov     al, bl
00004377 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:
00004378 8025[06770100]7F       <1>     and     byte [MOTOR_STATUS], 01111111b ; INDICATE A READ OPERATION
0000437F 66B846E6               <1>     mov     ax, 0E646h              ; AX = NEC COMMAND, DMA COMMAND
                                <1>     ;call   RD_WR_VF                ; COMMON READ/WRITE/VERIFY
                                <1>     ;retn
                                <1>     ; 06/08/2022
00004383 EB0B                   <1>     jmp     short RD_WR_VF
                                <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:
00004385 66B84AC5               <1>     mov     ax, 0C54Ah              ; AX = NEC COMMAND, DMA COMMAND
00004389 800D[06770100]80       <1>     or      byte [MOTOR_STATUS], 10000000b ; INDICATE WRITE OPERATION
                                <1>     ;call   RD_WR_VF                ; COMMON READ/WRITE/VERIFY
                                <1>     ;retn
                                <1>     ; 06/08/2022
                                <1>     ;jmp    short RD_WR_VF
                                <1>
                                <1>     ; 09/08/2022
                                <1>     ; 06/08/2022 - TRDOS 386 Kernel v2.0.5
                                <1> ;--------------------------------------------------------------------------------
                                <1> ; RD_WR_VF
                                <1> ;   COMMON READ, WRITE AND VERIFY:
                                <1> ;   MAIN LOOP FOR STATE RETRIES.
                                <1> ;
                                <1> ; ON ENTRY: AH = READ/WRITE/VERIFY NEC PARAMETER
                                <1> ;           AL = READ/WRITE/VERIFY DMA PARAMETER
                                <1> ;
                                <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
                                <1> ;--------------------------------------------------------------------------------
                                <1> RD_WR_VF:
                                <1>     ; 02/12/2023
                                <1>     ; 11/08/2022
                                <1>     ; 09/08/2022
                                <1>     ; 07/08/2022
                                <1>     ; 06/08/2022
                                <1>     ; 11/04/2021 (32 bit push/pop, AX -> EAX)
                                <1>
00004390 50                     <1>     push    eax                     ; SAVE DMA, NEC PARAMETERS
                                <1>
                                <1>     ; 02/12/2023
                                <1>     ; (diskette change check for 'dir', 28 seconds)
00004391 A0[01770100]           <1>     mov     al, [TIMER_LOW+1]
00004396 D0E8                   <1>     shr     al, 1
00004398 A2[667E0100]           <1>     mov     [P_TIMER], al
                                <1>
0000439D E803040000             <1>     call    XLAT_NEW                ; TRANSLATE STATE TO PRESENT ARCH.
000043A2 E86A040000             <1>     call    SETUP_STATE             ; INITIALIZE START AND END RATE
000043A7 58                     <1>     pop     eax                     ; RESTORE READ/WRITE/VERIFY
                                <1> DO_AGAIN:
000043A8 50                     <1>     push    eax                     ; SAVE READ/WRITE/VERIFY PARAMETER
```

```
000043A9 E8F8040000          <1>     call    MED_CHANGE              ; MEDIA CHANGE AND RESET IF CHANGED
000043AE 58                  <1>     pop     eax                     ; RESTORE READ/WRITE/VERIFY
                             <1>     ;jc     short RWV_END           ; MEDIA CHANGE ERROR OR TIME-OUT
                             <1>     ; 07/08/2022
000043AF 7305                <1>     jnc     short RWV
000043B1 E9BC000000          <1>     jmp     RWV_END
                             <1> RWV:
000043B6 50                  <1>     push    eax                     ; SAVE READ/WRITE/VERIFY PARAMETER
000043B7 8AB7[13770100]      <1>     mov     dh, [DSK_STATE+edi]     ; GET RATE STATE OF THIS DRIVE
000043BD 80E6C0              <1>     and     dh, RATE_MSK            ; KEEP ONLY RATE
000043C0 E83E080000          <1>     call    CMOS_TYPE               ; RETURN DRIVE TYPE IN AL (AL)
                             <1>     ;;20/02/2015
                             <1>     ;;jc    short RWV_ASSUME        ; ERROR IN CMOS
000043C5 7447                <1>     jz      short RWV_ASSUME ; 20/02/2015
000043C7 3C01                <1>     cmp     al, 1                   ; 40 TRACK DRIVE?
000043C9 750D                <1>     jne     short RWV_1             ; NO, BYPASS CMOS VALIDITY CHECK
000043CB F687[13770100]01    <1>     test    byte [DSK_STATE+edi], TRK_CAPA ; CHECK FOR 40 TRACK DRIVE
000043D2 7411                <1>     jz      short RWV_2             ; YES, CMOS IS CORRECT
                             <1>     ;mov    al, 2                   ; CHANGE TO 1.2M
                             <1>     ; 06/08/2022
000043D4 FEC0                <1>     inc     al ; al = 2
000043D6 EB0D                <1>     jmp     short RWV_2
                             <1> RWV_1:
                             <1>     ; 09/08/2022
                             <1>     ;jb     short RWV_2             ; NO DRIVE SPECIFIED, CONTINUE
000043D8 F687[13770100]01    <1>     test    byte [DSK_STATE+edi], TRK_CAPA ; IS IT REALLY 40 TRACK?
000043DF 7504                <1>     jnz     short RWV_2             ; NO, 80 TRACK
000043E1 B001                <1>     mov     al, 1                   ; IT IS 40 TRACK, FIX CMOS VALUE
000043E3 EB00                <1>     jmp     short rwv_3
                             <1> RWV_2:
                             <1>     ; 09/08/2022
                             <1>     ;or     al, al                  ; TEST FOR NO DRIVE
                             <1>     ;jz     short RWV_ASSUME        ; ASSUME TYPE, USE MAX TRACK
                             <1> rwv_3:
000043E5 E855030000          <1>     call    DR_TYPE_CHECK           ; RTN CS:BX = MEDIA/DRIVE PARAM TBL.
000043EA 7222                <1>     jc      short RWV_ASSUME        ; TYPE NOT IN TABLE (BAD CMOS)
                             <1>
                             <1> ;-----      SEARCH FOR MEDIA/DRIVE PARAMETER TABLE
                             <1>
000043EC 57                  <1>     push    edi                     ; SAVE DRIVE #
                             <1>     ; 09/08/2022
                             <1>     ;xor    ebx, ebx                ; EBX = INDEX TO DR_TYPE TABLE
000043ED BB[F8650000]        <1>     mov     ebx, DR_TYPE
                             <1>     ;mov    ecx, DR_CNT             ; ECX = LOOP COUNT
000043F2 B106                <1>     mov     cl, DR_CNT
                             <1> RWV_DR_SEARCH:
                             <1>     ;mov    ah, [DR_TYPE+ebx]       ; GET DRIVE TYPE
000043F4 8A23                <1>     mov     ah, [ebx]
000043F6 80E47F              <1>     and     ah, BIT7OFF             ; MASK OUT MSB
000043F9 38E0                <1>     cmp     al, ah                  ; DRIVE TYPE MATCH?
                             <1>     ; 09/08/2022
                             <1>     ;cmp    dl, ah
000043FB 7509                <1>     jne     short RWV_NXT_MD        ; NO, CHECK NEXT DRIVE TYPE
                             <1> RWV_DR_FND:
                             <1>     ;mov    edi, [DR_TYPE+ebx+1]    ; EDI = MEDIA/DRIVE PARAMETER TABLE
000043FD 43                  <1>     inc     ebx
000043FE 8B3B                <1>     mov     edi, [ebx]
00004400 4B                  <1>     dec     ebx
                             <1> RWV_MD_SEARH:
00004401 3A770C              <1>         cmp dh, [edi+MD.RATE]       ; MATCH?
00004404 741D                <1>     je      short RWV_MD_FND        ; YES, GO GET 1ST SPECIFY BYTE
                             <1> RWV_NXT_MD:
00004406 83C305              <1>     add     ebx, 5                  ; CHECK NEXT DRIVE TYPE
                             <1>     ;loop   RWV_DR_SEARCH
00004409 FEC9                <1>     dec     cl
0000440B 75E7                <1>     jnz     short RWV_DR_SEARCH
0000440D 5F                  <1>     pop     edi                     ; RESTORE DRIVE #
                             <1>
                             <1> ;-----      ASSUME PRIMARY DRIVE IS INSTALLED AS SHIPPED
                             <1>
                             <1> RWV_ASSUME:
0000440E BB[16660000]        <1>     mov     ebx, MD_TBL1            ; POINT TO 40 TRACK 250 KBS
00004413 F687[13770100]01    <1>     test    byte [DSK_STATE+edi], TRK_CAPA ; TEST FOR 80 TRACK
0000441A 740A                <1>     jz      short RWV_MD_FND1       ; MUST BE 40 TRACK
0000441C BB[30660000]        <1>     mov     ebx, MD_TBL3            ; POINT TO 80 TRACK 500 KBS
00004421 EB03                <1>     jmp     short RWV_MD_FND1       ; GO SPECIFY PARAMTERS
                             <1>
                             <1> ;-----      CS:BX POINTS TO MEDIA/DRIVE PARAMETER TABLE
                             <1>
                             <1> RWV_MD_FND:
00004423 89FB                <1>     mov     ebx, edi                ; BX = MEDIA/DRIVE PARAMETER TABLE
00004425 5F                  <1>     pop     edi                     ; RESTORE DRIVE #
                             <1>
                             <1> ;-----      SEND THE SPECIFY COMMAND TO THE CONTROLLER
                             <1>
                             <1> RWV_MD_FND1:
00004426 E85A030000          <1>     call    SEND_SPEC_MD
0000442B E8E3040000          <1>     call    CHK_LASTRATE            ; ZF=1 ATTEMP RATE IS SAME AS LAST RATE
00004430 7405                <1>     jz      short RWV_DBL           ; YES,SKIP SEND RATE COMMAND
00004432 E8BC040000          <1>     call    SEND_RATE               ; SEND DATA RATE TO NEC
                             <1> RWV_DBL:
00004437 53                  <1>     push    ebx                     ; SAVE MEDIA/DRIVE PARAM TBL ADDRESS
00004438 E825070000          <1>     call    SETUP_DBL               ; CHECK FOR DOUBLE STEP
0000443D 5B                  <1>     pop     ebx                     ; RESTORE ADDRESS
0000443E 7225                <1>     jc      short CHK_RET           ; ERROR FROM READ ID, POSSIBLE RETRY
                             <1>     ;pop    eax                     ; RESTORE NEC COMMAND
                             <1>     ;push   eax                     ; SAVE NEC COMMAND
                             <1>     ; 09/08/2022
00004440 8B0424              <1>     mov     eax, [esp]
                             <1>     ;push   ebx                     ; SAVE MEDIA/DRIVE PARAM TBL ADDRESS
00004443 E8E0040000          <1>     call    DMA_SETUP               ; SET UP THE DMA
                             <1>     ;pop    ebx
00004448 58                  <1>     pop     eax                     ; RESTORE NEC COMMAND
00004449 7231                <1>     jc      short RWV_BAC           ; CHECK FOR DMA BOUNDARY ERROR
0000444B 50                  <1>     push    eax                     ; SAVE NEC COMMAND
0000444C 53                  <1>     push    ebx                     ; SAVE MEDIA/DRIVE PARAM TBL ADDRESS
                             <1>     ; 11/08/2022
0000444D 8B5C2420            <1>     mov     ebx, [esp+32] ; ECX
00004451 E866050000          <1>     call    NEC_INIT                ; INITIALIZE NEC
00004456 5B                  <1>     pop     ebx                     ; RESTORE ADDRESS
00004457 720C                <1>     jc      short CHK_RET           ; ERROR - EXIT
00004459 E88C050000          <1>     call    RWV_COM                 ; OP CODE COMMON TO READ/WRITE
0000445E 7205                <1>     jc      short CHK_RET           ; ERROR - EXIT
00004460 E8D2050000          <1>     call    NEC_TERM                ; TERMINATE, GET STATUS, ETC.
                             <1> CHK_RET:
00004465 E871060000          <1>     call    RETRY                   ; CHECK FOR, SETUP RETRY
0000446A 58                  <1>     pop     eax                     ; RESTORE READ/WRITE PARAMETER
0000446B 7305                <1>     jnc     short RWV_END           ; CY = 0 NO RETRY
0000446D E936FFFFFF          <1>         jmp DO_AGAIN                ; CY = 1 MEANS RETRY
                             <1> RWV_END:
00004472 E81C060000          <1>     call    DSTATE                  ; ESTABLISH STATE IF SUCCESSFUL
00004477 E8AE060000          <1>     call    NUM_TRANS               ; AL = NUMBER TRANSFERRED
                             <1> RWV_BAC:                            ; BAD DMA ERROR ENTRY
                             <1>     ; 06/08/2022
```

```
                        <1>     ;push   eax                     ; SAVE NUMBER TRANSFERRED
                        <1>     ;call   XLAT_OLD                ; TRANSLATE STATE TO COMPATIBLE MODE
                        <1>     ;pop    eax                     ; RESTORE NUMBER TRANSFERRED
                        <1>     ;call   SETUP_END               ; VARIOUS CLEANUPS
                        <1>     ;retn
                        <1>
                        <1> ;--------------------------------------------------------------------------------
                        <1> ; SETUP_END
                        <1> ;   RESTORES @MOTOR_COUNT TO PARAMETER PROVIDED IN TABLE
                        <1> ;   AND LOADS @DSKETTE_STATUS TO AH, AND SETS CY.
                        <1> ;
                        <1> ; ON EXIT:
                        <1> ;   AH, @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
                        <1> ;--------------------------------------------------------------------------------
                        <1> SETUP_END:
                        <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
                        <1>     ;mov    dl, 2                   ; GET THE MOTOR WAIT PARAMETER
                        <1>     ;push   ax                      ; SAVE NUMBER TRANSFERRED
                        <1>     ; 11/04/2021
0000447C 50             <1>     push    eax
                        <1>     ; 06/08/2022
0000447D B002           <1>     mov     al, 2                   ; GET THE MOTOR WAIT PARAMETER
0000447F E888070000     <1>     call    GET_PARM
00004484 8825[07770100] <1>     mov     [MOTOR_COUNT], ah       ; STORE UPON RETURN
                        <1>     ;pop    ax                      ; RESTORE NUMBER TRANSFERRED
                        <1>     ; 11/04/2021
0000448A 58             <1>     pop     eax
0000448B 8A25[08770100] <1>     mov     ah, [DSKETTE_STATUS]    ; GET STATUS OF OPERATION
00004491 08E4           <1>     or      ah, ah                  ; CHECK FOR ERROR
00004493 7403           <1>     jz      short NUN_ERR           ; NO ERROR
00004495 30C0           <1>     xor     al, al                  ; CLEAR NUMBER RETURNED
                        <1>     ; 06/08/2022
00004497 F9             <1>     stc
                        <1> NUN_ERR:
                        <1>     ;cmp    ah, 1                   ; SET THE CARRY FLAG TO INDICATE
                        <1>     ;cmc                            ; SUCCESS OR FAILURE
00004498 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:
00004499 8025[06770100]7F <1>     and     byte [MOTOR_STATUS], 01111111b ; INDICATE A READ OPERATION
000044A0 66B842E6       <1>     mov     ax, 0E642h              ; AX = NEC COMMAND, DMA COMMAND
                        <1>     ;call   RD_WR_VF                ; COMMON READ/WRITE/VERIFY
                        <1>     ;retn
                        <1>     ; 06/08/2022
000044A4 E9E7FEFFFF     <1>     jmp     RD_WR_VF
                        <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:
                        <1>     ; 11/08/2022
                        <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
000044A9 E8F7020000     <1>     call    XLAT_NEW                ; TRANSLATE STATE TO PRESENT ARCH.
000044AE E89F030000     <1>     call    FMT_INIT                ; ESTABLISH STATE IF UNESTABLISHED
000044B3 800D[06770100]80 <1>     or      byte [MOTOR_STATUS], 10000000b ; INDICATE WRITE OPERATION
000044BA E8E7030000     <1>     call    MED_CHANGE              ; CHECK MEDIA CHANGE AND RESET IF SO
000044BF 7261           <1>         jc      short FM_DON            ; MEDIA CHANGED, SKIP
000044C1 E896020000     <1>     call    SEND_SPEC               ; SEND SPECIFY COMMAND TO NEC
000044C6 E848040000     <1>     call    CHK_LASTRATE            ; ZF=1 ATTEMPT RATE IS SAME AS LAST RATE
000044CB 7405           <1>         jz      short FM_WR             ; YES, SKIP SPECIFY COMMAND
000044CD E821040000     <1>     call    SEND_RATE               ; SEND DATA RATE TO CONTROLLER
                        <1> FM_WR:
000044D2 E8C8040000     <1>     call    FMTDMA_SET              ; SET UP THE DMA FOR FORMAT
000044D7 7249           <1>         jc      short FM_DON            ; RETURN WITH ERROR
000044D9 B44D           <1>     mov     ah, 04Dh                ; ESTABLISH THE FORMAT COMMAND
                        <1>     ; 11/08/2022
000044DB 8B5C2418       <1>     mov     ebx, [esp+24] ; ECX
000044DF E8D8040000     <1>     call    NEC_INIT                ; INITIALIZE THE NEC
000044E4 723C           <1>     jc      short FM_DON              ; ERROR - EXIT
000044E6 B8[22450000]   <1>         mov     eax, FM_DON             ; LOAD ERROR ADDRESS
000044EB 50             <1>     push    eax                     ; PUSH NEC_OUT ERROR RETURN
                        <1>     ;mov    dl, 3                   ; BYTES/SECTOR VALUE TO NEC
                        <1>     ; 06/08/2022
000044EC B003           <1>     mov     al, 3
000044EE E819070000     <1>     call    GET_PARM
000044F3 E80A080000     <1>     call    NEC_OUTPUT
                        <1>     ;mov    dl, 4                   ; SECTORS/TRACK VALUE TO NEC
                        <1>     ; 06/08/2022
000044F8 B004           <1>     mov     al, 4
000044FA E80D070000     <1>     call    GET_PARM
000044FF E8FE070000     <1>     call    NEC_OUTPUT
                        <1>     ;mov    dl, 7                   ; GAP LENGTH VALUE TO NEC
                        <1>     ; 06/08/2022
00004504 B007           <1>     mov     al, 7
00004506 E801070000     <1>     call    GET_PARM
0000450B E8F2070000     <1>     call    NEC_OUTPUT
                        <1>     ;mov    dl, 8                   ; FILLER BYTE TO NEC
                        <1>     ; 06/08/2022
00004510 B008           <1>     mov     al, 8
00004512 E8F5060000     <1>     call    GET_PARM
00004517 E8E6070000     <1>     call    NEC_OUTPUT
0000451C 58             <1>     pop     eax                     ; THROW AWAY ERROR
0000451D E815050000     <1>     call    NEC_TERM                ; TERMINATE, RECEIVE STATUS, ETC,
                        <1> FM_DON:
                        <1>     ; 06/08/2022
                        <1>     ;call   XLAT_OLD                ; TRANSLATE STATE TO COMPATIBLE MODE
                        <1>     ; 06/08/2022
00004522 E947FEFFFF     <1>     jmp     SETUP_END_X
```

```
1106                            <1>     ;call   SETUP_END              ; VARIOUS CLEANUPS
1107                            <1>     ;; 06/08/2022
1108                            <1>     ;mov    ebx, esi               ; GET SAVED AL TO BL
1109                            <1>     ;mov    al, bl                 ; PUT BACK FOR RETURN
1110                            <1>     ;retn
1111                            <1>
1112                            <1> ;--------------------------------------------------------------------------
1113                            <1> ; FNC_ERR
1114                            <1> ;   INVALID FUNCTION REQUESTED OR INVALID DRIVE:
1115                            <1> ;   SET BAD COMMAND IN STATUS.
1116                            <1> ;
1117                            <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
1118                            <1> ;--------------------------------------------------------------------------
1119                            <1> FNC_ERR:                            ; INVALID FUNCTION REQUEST
1120                            <1>     ;mov    ax, si
1121                            <1>     ; 06/08/2022
1122 00004527 89F0             <1>     mov     eax, esi               ; RESTORE AL
1123 00004529 B401             <1>     mov     ah, BAD_CMD            ; SET BAD COMMAND ERROR
1124 0000452B 8825[08770100]   <1>     mov     [DSKETTE_STATUS], ah   ; STORE IN DATA AREA
1125 00004531 F9               <1>     stc                            ; SET CARRY INDICATING ERROR
1126 00004532 C3               <1>     retn
1127                            <1>
1128                            <1> ; 06/08/2022 - TRDOS 386 Kernel v2.0.5
1129                            <1> ; 30/08/2020
1130                            <1> ; 29/08/2020
1131                            <1> ; 01/06/2016
1132                            <1> ; 28/05/2016
1133                            <1> ; 27/05/2016 - TRDOS 386 (TRDOS v.2.0)
1134                            <1> ;--------------------------------------------------------------------------
1135                            <1> ; DISK_PARMS       (AH = 08H)
1136                            <1> ;   READ DRIVE PARAMETERS.
1137                            <1> ;
1138                            <1> ; ON ENTRY: DI : DRIVE #
1139                            <1> ;           ; 27/05/2016
1140                            <1> ;           EBX = Buffer Address for floppy disk parameters table (16 bytes)
1141                            <1> ;
1142                            <1> ; ON EXIT:  CL/[BP]   = BITS 7 & 6 HI 2 BITS OF MAX CYLINDER
1143                            <1> ;                       BITS 0-5 MAX SECTORS/TRACK
1144                            <1> ;           CH/[BP+1] = LOW 8 BITS OF MAX CYLINDER
1145                            <1> ;           BL/[BP+2] = BITS 7-4 = 0
1146                            <1> ;                       BITS 3-0 = VALID CMOS DRIVE TYPE
1147                            <1> ;           BH/[BP+3] = 0
1148                            <1> ;           DL/[BP+4] = # DRIVES INSTALLED (VALUE CHECKED)
1149                            <1> ;           DH/[BP+5] = MAX HEAD #
1150                            <1> ;         ** 27/05/2016 - TRDOS 386 (TRDOS v2.0) **
1151                            <1> ;            ** EBX = Buffer address for floppy disk parameters table **
1152                            <1> ;           ;DI/[BP+6] = OFFSET TO DISK_BASE
1153                            <1> ;           ;ES        = SEGMENT OF DISK_BASE
1154                            <1> ;
1155                            <1> ;           AX        = 0
1156                            <1> ;
1157                            <1> ;           NOTE : THE ABOVE INFORMATION IS STORED IN THE USERS STACK AT
1158                            <1> ;                  THE LOCATIONS WHERE THE MAIN ROUTINE WILL POP THEM
1159                            <1> ;                  INTO THE APPROPRIATE REGISTERS BEFORE RETURNING TO THE
1160                            <1> ;                  CALLER.
1161                            <1> ;--------------------------------------------------------------------------
1162                            <1> DSK_PARMS:
1163                            <1>     ; 09/08/2022
1164                            <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
1165                            <1>     ;
1166                            <1>     ; Registers on stack:
1167                            <1>     ;    ebx = esp+28
1168                            <1>     ;    ecx = esp+24
1169                            <1>     ;    edx = esp+20
1170                            <1>     ;    esi = esp+16
1171                            <1>     ;    edi = esp+12
1172                            <1>     ;    return address from DSKETTE_IO_1 = esp+8
1173                            <1>     ;    ebp = esp+4
1174                            <1>     ;    return address from DSK_PARMS = esp
1175                            <1>     ;
1176                            <1>     ; INPUT:
1177                            <1>     ;   ebp = buffer address
1178                            <1>     ;   edi = drive number (0 or 1)
1179                            <1>
1180                            <1>     ; OUTPUT:
1181                            <1>     ;   ebx = [esp+28] ((BL = cmos type, BH = 0))
1182                            <1>     ;   ecx = [esp+24] ((CL = sectors per track, CH = tracks - 1))
1183                            <1>     ;   edx = [esp+20] ((DL = floppy drive count, DH = heads - 1))
1184                            <1>     ;   user's buffer = FDPT table
1185                            <1>
1186 00004533 E86D020000       <1>     call    XLAT_NEW               ; TRANSLATE STATE TO PRESENT ARCH,
1187                            <1>
1188                            <1>     ;;mov   word [bp+2], 0         ; DRIVE TYPE = 0
1189                            <1>             ;;mov   ax, [EQUIP_FLAG]       ; LOAD EQUIPMENT FLAG FOR # DISKETTES
1190                            <1>             ;;and   al, 11000001b          ; KEEP DISKETTE DRIVE BITS
1191                            <1>             ;;mov   dl, 2                  ; DISKETTE DRIVES = 2
1192                            <1>             ;;cmp   al, 01000001b          ; 2 DRIVES INSTALLED ?
1193                            <1>             ;;jz    short STO_DL           ; IF YES JUMP
1194                            <1>             ;;dec   dl                     ; DISKETTE DRIVES = 1
1195                            <1>             ;;cmp   al, 00000001b          ; 1 DRIVE INSTALLED ?
1196                            <1>             ;;jnz   short NON_DRV          ; IF NO JUMP
1197                            <1>
1198 00004538 29D2             <1>     sub     edx, edx
1199 0000453A 66A1[7C660000]   <1>     mov     ax, [fd0_type]
1200 00004540 6621C0           <1>     and     ax, ax
1201 00004543 747C             <1>         jz      short NON_DRV
1202 00004545 FEC2             <1>     inc     dl
1203 00004547 20E4             <1>     and     ah, ah
1204 00004549 7402             <1>     jz      short STO_DL
1205 0000454B FEC2             <1>     inc     dl
1206                            <1> STO_DL:
1207                            <1>     ; 30/08/2020
1208                            <1>     ;cmp    dx, di
1209                            <1>     ; 06/08/2022
1210 0000454D 39FA             <1>     cmp     edx, edi
1211 0000454F 7670             <1>     jna     short NON_DRV
1212                            <1>     ;
1213                            <1>     ;;mov   [bp+4], dl             ; STORE NUMBER OF DRIVES
1214                            <1>     ;mov    [ebp+8], edx ; 20/02/2015
1215                            <1>
1216                            <1>     ; 06/08/2022
1217 00004551 FEC6             <1>     inc     dh ; number of heads - 1
1218                            <1>     ; dh = 1
1219                            <1>     ; dl = number of floppy/diskette drives (DL)
1220 00004553 89542414         <1>     mov     [esp+20], edx
1221                            <1>
1222                            <1>     ; 11/04/2021
1223                            <1>     ;cmp    di, 1                  ; CHECK FOR VALID DRIVE
1224                            <1>     ;;ja    short NON_DRV1         ; DRIVE INVALID
1225                            <1>     ;ja     NON_DRV1 ; 29/08/2020
1226                            <1>     ;
1227                            <1>     ;;mov   byte [bp+5], 1         ; MAXIMUM HEAD NUMBER =      1
1228                            <1>     ; 06/08/2022
1229                            <1>     ;mov    byte [ebp+9], 1 ; 20/02/2015
```

```
1230                                    <1>
1231 00004557 E8A7060000                <1>     call    CMOS_TYPE               ; RETURN DRIVE TYPE IN AL
1232                                    <1>     ;;20/02/2015
1233                                    <1>     ;;jc    short CHK_EST           ; IF CMOS BAD CHECKSUM ESTABLISHED
1234                                    <1>     ;;or    al, al                  ; TEST FOR NO DRIVE TYPE
1235 0000455C 740F                      <1>     jz      short CHK_EST           ; JUMP IF SO
1236 0000455E E8DC010000                <1>     call    DR_TYPE_CHECK           ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
1237 00004563 7208                      <1>     jc      short CHK_EST           ; TYPE NOT IN TABLE (POSSIBLE BAD CMOS)
1238                                    <1>     ;mov    [bp+2], al              ; STORE VALID CMOS DRIVE TYPE
1239                                    <1>         ;mov       [ebp+4], al ; 06/02/2015
1240 00004565 8A4B04                    <1>     mov     cl, [ebx+MD.SEC_TRK]    ; GET SECTOR/TRACK
1241 00004568 8A6B0B                    <1>         mov ch, [ebx+MD.MAX_TRK]    ; GET MAX. TRACK NUMBER
1242 0000456B EB36                      <1>     jmp     short STO_CX            ; CMOS GOOD, USE CMOS
1243                                    <1> CHK_EST:
1244 0000456D 8AA7[13770100]            <1>     mov     ah, [DSK_STATE+edi]     ; LOAD STATE FOR THIS DRIVE
1245 00004573 F6C410                    <1>     test    ah, MED_DET             ; CHECK FOR ESTABLISHED STATE
1246 00004576 744D                      <1>     jz      short NON_DRV1          ; CMOS BAD/INVALID OR UNESTABLISHED
1247                                    <1> USE_EST:
1248 00004578 80E4C0                    <1>     and     ah, RATE_MSK            ; ISOLATE STATE
1249 0000457B 80FC80                    <1>     cmp     ah, RATE_250            ; RATE 250 ?
1250 0000457E 755C                      <1>     jne     short USE_EST2          ; NO, GO CHECK OTHER RATE
1251                                    <1>
1252                                    <1> ;-----      DATA RATE IS 250 KBS, TRY 360 KB TABLE FIRST
1253                                    <1>
1254 00004580 B001                      <1>     mov     al, 1                   ; DRIVE TYPE 1 (360KB)
1255 00004582 E8B8010000                <1>     call    DR_TYPE_CHECK           ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
1256 00004587 8A4B04                    <1>         mov cl, [ebx+MD.SEC_TRK]    ; GET SECTOR/TRACK
1257 0000458A 8A6B0B                    <1>         mov ch, [ebx+MD.MAX_TRK]    ; GET MAX. TRACK NUMBER
1258 0000458D F687[13770100]01          <1>     test    byte [DSK_STATE+edi], TRK_CAPA ; 80 TRACK ?
1259 00004594 740D                      <1>     jz      short STO_CX            ; MUST BE 360KB DRIVE
1260                                    <1>
1261                                    <1> ;-----      IT IS 1.44 MB DRIVE
1262                                    <1>
1263                                    <1> PARM144:
1264 00004596 B004                      <1>     mov     al, 4                   ; DRIVE TYPE 4 (1.44MB)
1265 00004598 E8A2010000                <1>     call    DR_TYPE_CHECK           ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
1266 0000459D 8A4B04                    <1>     mov     cl, [ebx+MD.SEC_TRK]    ; GET SECTOR/TRACK
1267 000045A0 8A6B0B                    <1>     mov     ch, [ebx+MD.MAX_TRK]    ; GET MAX. TRACK NUMBER
1268                                    <1> STO_CX:
1269                                    <1>     ;mov    [ebp], ecx              ; SAVE POINTER IN STACK FOR RETURN
1270                                    <1>     ; 06/08/2022
1271 000045A3 894C2418                  <1>     mov     [esp+24], ecx ; spt (cl), tracks - 1 (ch)
1272                                    <1> ES_DI:
1273                                    <1>     ;mov    [bp+6], bx              ; ADDRESS OF MEDIA/DRIVE PARM TABLE
1274                                    <1>     ;mov    [ebp+12], ebx ; 06/02/2015
1275                                    <1>     ;mov    ax, cs                  ; SEGMENT MEDIA/DRIVE PARAMETER TABLE
1276                                    <1>     ;mov    es, ax                  ; ES IS SEGMENT OF TABLE
1277                                    <1>     ;
1278                                    <1>     ; 28/05/2016
1279                                    <1>     ; 27/05/2016
1280                                    <1>     ; return floppy disk parameters table to user
1281                                    <1>     ; in user's buffer, which is pointed by EBX
1282                                    <1>
1283                                    <1>     ; 09/08/2022
1284                                    <1>     ;movzx  eax, al
1285 000045A7 29C9                      <1>     sub     ecx, ecx
1286 000045A9 88C1                      <1>     mov     cl, al
1287                                    <1>     ;;mov   [ebp+4], eax  ; ebx   ; drive type (for floppy drives)
1288                                    <1>     ; 06/08/2022
1289                                    <1>     ;mov    [esp+28], eax ; drive type
1290                                    <1>     ; 09/08/2022
1291 000045AB 894C241C                  <1>     mov     [esp+28], ecx ; drive type
1292                                    <1>
1293                                    <1>     ; 06/08/2022
1294                                    <1>     ;push   edi
1295                                    <1>
1296                                    <1>     ;mov    edi, [ebp+4]            ; ebx (input), user's buffer address
1297                                    <1>     ; 06/08/2022
1298                                    <1>     ;mov    edi, ebp ; [esp+28] ; user's buffer address
1299                                    <1>     ;
1300                                    <1>     ;; 29/08/2020
1301                                    <1>     ;or     edi, edi
1302                                    <1>     ;jz     short no_copy_fdpt
1303                                    <1>     ; 06/08/2022
1304 000045AF 09ED                      <1>     or      ebp, ebp ; [esp+28] = ebx ; user's buffer address if > 0
1305 000045B1 740B                      <1>     jz      short DP_OUT
1306                                    <1>     ; 06/08/2022
1307                                    <1>     ;push   edi
1308 000045B3 89EF                      <1>     mov     edi, ebp
1309                                    <1>     ;
1310                                    <1>     ; 06/08/2022
1311                                    <1>     ; 01/06/2016 (Int 33h, disk type return for floppy disks, in BL)
1312                                    <1>     ;mov    [user_buffer], eax    ; 01/06/2016 (overwrite ebx return value)
1313                                    <1>
1314                                    <1>     ;(Int 33h, Function 08h will replace user's buffer addr with disk type!)
1315                                    <1>     ;
1316 000045B5 89DE                      <1>     mov     esi, ebx                ; floppy disk parameter table (16 bytes)
1317                                    <1>     ;mov    ecx, 16 ; 16 bytes
1318                                    <1>         ; 09/08/2022
1319                                    <1>     ; ecx < 256
1320                                    <1>     ; 06/08/2022
1321                                    <1>     ;sub    ecx, ecx
1322 000045B7 B110                      <1>     mov     cl, 16
1323 000045B9 E82CC50000                <1>     call    transfer_to_user_buffer ; trdosk6.s (16/05/2016)
1324                                    <1> no_copy_fdpt:
1325                                    <1>     ; 06/08/2022
1326                                    <1>     ;pop    edi
1327                                    <1> DP_OUT:
1328                                    <1>     ; 06/08/2022
1329                                    <1>     ;call   XLAT_OLD                ; TRANSLATE STATE TO COMPATIBLE MODE
1330                                    <1>     ;xor    ax, ax                  ; CLEAR
1331                                    <1>     ; 06/08/2022
1332 000045BE 31C0                      <1>     xor     eax, eax
1333                                    <1>     ;clc
1334 000045C0 C3                        <1>     retn
1335                                    <1>
1336                                    <1> ;-----      NO DRIVE PRESENT HANDLER
1337                                    <1>
1338                                    <1> NON_DRV:
1339                                    <1>     ;;mov   byte [bp+4], 0          ; CLEAR NUMBER OF DRIVES
1340                                    <1>     ;mov    [ebp+8], edx ; 0 ; 20/02/2015
1341                                    <1>     ; 06/08/2022
1342 000045C1 89542414                  <1>     mov     [esp+20], edx ; 0 ; number of floppy drives and heads are 0
1343                                    <1> NON_DRV1:
1344 000045C5 6681FF8000                <1>     cmp     di, 80h                 ; CHECK FOR FIXED MEDIA TYPE REQUEST
1345 000045CA 7206                      <1>     jb      short NON_DRV2          ; CONTINUE IF NOT REQUEST FALL THROUGH
1346                                    <1>
1347                                    <1> ;-----      FIXED DISK REQUEST FALL THROUGH ERROR
1348                                    <1>
1349                                    <1>     ; 06/08/2022
1350                                    <1>     ;call   XLAT_OLD                ; ELSE TRANSLATE TO COMPATIBLE MODE
1351                                    <1>     ;mov    ax, si                  ; RESTORE AL
1352                                    <1>     ; 06/08/2022
1353 000045CC 89F0                      <1>     mov     eax, esi
```

```
1354 000045CE B401              <1>     mov     ah, BAD_CMD             ; SET BAD COMMAND ERROR
1355 000045D0 F9                <1>     stc
1356 000045D1 C3                <1>     retn
1357                            <1>
1358                            <1> NON_DRV2:
1359                            <1>     ;xor    ax, ax                  ; CLEAR PARMS IF NO DRIVES OR CMOS BAD
1360 000045D2 31C0              <1>     xor     eax, eax
1361                            <1>     ;mov    [ebp], ax               ; TRACKS, SECTORS/TRACK = 0
1362                            <1>     ; 06/08/2022
1363 000045D4 89442418          <1>     mov     [esp+24], eax ; spt and max. track number is 0
1364                            <1>
1365                            <1>     ;;mov   [bp+5], ah              ; HEAD = 0
1366                            <1>     ; 06/08/2022
1367                            <1>     ;mov    [ebp+9], ah ; 06/02/2015
1368                            <1>     ; 06/08/2022
1369                            <1>     ;mov    [esp+21], ah ; 0
1370                            <1>
1371                            <1>     ;;mov   [bp+6], ax              ; OFFSET TO DISK_BASE = 0
1372                            <1>     ; 06/08/2022
1373                            <1>     ;mov    [ebp+12], eax
1374                            <1>
1375                            <1>     ;;mov   es, ax                  ; ES IS SEGMENT OF TABLE
1376                            <1>     ;jmp    short DP_OUT
1377                            <1>
1378                            <1>     ; 06/08/2022
1379                            <1>     ; 30/08/2020
1380                            <1>     ;call   XLAT_OLD
1381                            <1>     ;;mov   ah, NOT_RDY ; drive not ready
1382 000045D8 B407              <1>     mov     ah, INIT_FAIL ; DRIVE PARAMETER ACTIVITY FAILED
1383 000045DA F9                <1>     stc     ; cf -> 1, ah = 'drive not ready' error code
1384 000045DB C3                <1>     retn
1385                            <1>
1386                            <1> ;-----      DATA RATE IS EITHER 300 KBS OR 500 KBS, TRY 1.2 MB TABLE FIRST
1387                            <1>
1388                            <1> USE_EST2:
1389 000045DC B002              <1>     mov     al, 2                   ; DRIVE TYPE 2 (1.2MB)
1390 000045DE E85C010000        <1>     call    DR_TYPE_CHECK           ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
1391 000045E3 8A4B04            <1>         mov     cl, [ebx+MD.SEC_TRK]    ; GET SECTOR/TRACK
1392 000045E6 8A6B0B            <1>         mov     ch, [ebx+MD.MAX_TRK]    ; GET MAX. TRACK NUMBER
1393 000045E9 80FC40            <1>     cmp     ah, RATE_300            ; RATE 300 ?
1394 000045EC 74B5              <1>     jz      short STO_CX            ; MUST BE 1.2MB DRIVE
1395 000045EE EBA6              <1>     jmp     short PARM144           ; ELSE, IT IS 1.44MB DRIVE
1396                            <1>
1397                            <1> ; 30/08/2020
1398                            <1>
1399                            <1> ;--------------------------------------------------------------------------------
1400                            <1> ; DISK_TYPE (AH = 15H)
1401                            <1> ;   THIS ROUTINE RETURNS THE TYPE OF MEDIA INSTALLED.
1402                            <1> ;
1403                            <1> ;  ON ENTRY:       DI = DRIVE #
1404                            <1> ;
1405                            <1> ;  ON EXIT: AH = DRIVE TYPE, CY=0
1406                            <1> ;--------------------------------------------------------------------------------
1407                            <1> DSK_TYPE:
1408                            <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
1409 000045F0 E8B0010000        <1>     call    XLAT_NEW                ; TRANSLATE STATE TO PRESENT ARCH.
1410 000045F5 8A87[13770100]    <1>     mov     al, [DSK_STATE+edi]     ; GET PRESENT STATE INFORMATION
1411 000045FB 08C0              <1>     or      al, al                  ; CHECK FOR NO DRIVE
1412 000045FD 740D              <1>     jz      short NO_DRV
1413 000045FF B401              <1>     mov     ah, NOCHGLN             ; NO CHANGE LINE FOR 40 TRACK DRIVE
1414 00004601 A801              <1>     test    al, TRK_CAPA            ; IS THIS DRIVE AN 80 TRACK DRIVE?
1415 00004603 7402              <1>     jz      short DT_BACK           ; IF NO JUMP
1416 00004605 B402              <1>     mov     ah, CHGLN               ; CHANGE LINE FOR 80 TRACK DRIVE
1417                            <1> DT_BACK:
1418                            <1>     ; 06/08/2022
1419                            <1>     ;;push  ax                      ; SAVE RETURN VALUE
1420                            <1>     ; 11/04/2021
1421                            <1>     ;push   eax
1422                            <1>     ;call   XLAT_OLD                ; TRANSLATE STATE TO COMPATIBLE MODE
1423                            <1>     ;; 11/04/2021
1424                            <1>     ;pop    eax
1425                            <1>     ;; pop  ax
1426                            <1>     ;clc                            ; NO ERROR
1427                            <1>     ;mov    bx, si                  ; GET SAVED AL TO BL
1428                            <1>     ; 06/08/2022
1429 00004607 89F3              <1>     mov     ebx, esi
1430 00004609 88D8              <1>     mov     al, bl                  ; PUT BACK FOR RETURN
1431 0000460B C3                <1>     retn
1432                            <1> NO_DRV:
1433                            <1>     ;xor    ah, ah                  ; NO DRIVE PRESENT OR UNKNOWN
1434                            <1>     ;jmp    short DT_BACK
1435                            <1>
1436                            <1>     ; 06/08/2022
1437                            <1>     ; 30/08/2020
1438                            <1>     ;call   XLAT_OLD
1439 0000460C 29C0              <1>     sub     eax, eax
1440 0000460E F9                <1>     stc     ; cf = 1 -> drive not ready, ah = 0 (disk type = 0)
1441 0000460F C3                <1>     retn
1442                            <1>
1443                            <1> ;--------------------------------------------------------------------------------
1444                            <1> ; DISK_CHANGE      (AH = 16H)
1445                            <1> ;   THIS ROUTINE RETURNS THE STATE OF THE DISK CHANGE LINE.
1446                            <1> ;
1447                            <1> ; ON ENTRY: DI = DRIVE #
1448                            <1> ;
1449                            <1> ; ON EXIT:  AH = @DSKETTE_STATUS
1450                            <1> ;               00 - DISK CHANGE LINE INACTIVE, CY = 0
1451                            <1> ;               06 - DISK CHANGE LINE ACTIVE, CY = 1
1452                            <1> ;--------------------------------------------------------------------------------
1453                            <1> DSK_CHANGE:
1454                            <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
1455 00004610 E890010000        <1>     call    XLAT_NEW                ; TRANSLATE STATE TO PRESENT ARCH.
1456 00004615 8A87[13770100]    <1>     mov     al, [DSK_STATE+edi]     ; GET MEDIA STATE INFORMATION
1457 0000461B 08C0              <1>     or      al, al                  ; DRIVE PRESENT ?
1458 0000461D 7417              <1>     jz      short DC_NON            ; JUMP IF NO DRIVE
1459 0000461F A801              <1>     test    al, TRK_CAPA            ; 80 TRACK DRIVE ?
1460 00004621 7407              <1>     jz      short SETIT             ; IF SO , CHECK CHANGE LINE
1461                            <1> DC0:
1462 00004623 E86B080000        <1>         call    READ_DSKCHNG            ; GO CHECK STATE OF DISK CHANGE LINE
1463 00004628 7407              <1>     jz      short FINIS             ; CHANGE LINE NOT ACTIVE
1464                            <1> SETIT:
1465 0000462A C605[08770100]06  <1>     mov     byte [DSKETTE_STATUS], MEDIA_CHANGE ; INDICATE MEDIA REMOVED
1466                            <1>
1467                            <1> FINIS:      ; 06/08/2022
1468                            <1>     ;call   XLAT_OLD                ; TRANSLATE STATE TO COMPATIBLE MODE
1469                            <1>     ; 06/08/2022
1470 00004631 E938FDFFFF        <1>     jmp     SETUP_END_X
1471                            <1>     ;call   SETUP_END               ; VARIOUS CLEANUPS
1472                            <1>     ;; 06/08/2022
1473                            <1>     ;mov    ebx, esi                ; GET SAVED AL TO BL
1474                            <1>     ;mov    al, bl                  ; PUT BACK FOR RETURN
1475                            <1>     ;retn
1476                            <1> DC_NON:
1477 00004636 800D[08770100]80  <1>     or      byte [DSKETTE_STATUS], TIME_OUT ; SET TIMEOUT, NO DRIVE
```

```
1478 0000463D EBF2                <1>     jmp     short FINIS
1479                              <1>
1480                              <1> ;--------------------------------------------------------------------------------
1481                              <1> ; FORMAT_SET       (AH = 17H)
1482                              <1> ;   THIS ROUTINE IS USED TO ESTABLISH THE TYPE OF MEDIA TO BE USED
1483                              <1> ;   FOR THE FOLLOWING FORMAT OPERATION.
1484                              <1> ;
1485                              <1> ; ON ENTRY: SI LOW = DASD TYPE FOR FORMAT
1486                              <1> ;           DI     = DRIVE #
1487                              <1> ;
1488                              <1> ; ON EXIT:  @DSKETTE_STATUS REFLECTS STATUS
1489                              <1> ;           AH = @DSKETTE_STATUS
1490                              <1> ;           CY = 1 IF ERROR
1491                              <1> ;--------------------------------------------------------------------------------
1492                              <1> FORMAT_SET:
1493 0000463F E861010000          <1>     call    XLAT_NEW                ; TRANSLATE STATE TO PRESENT ARCH.
1494                              <1>     ;push   si                      ; SAVE DASD TYPE
1495                              <1>     ; 11/04/2021
1496 00004644 56                  <1>     push    esi
1497                              <1>     ;mov    ax, si                  ; AH = ? , AL = DASD TYPE
1498                              <1>     ;xor    ah, ah                  ; AH = 0 , AL = DASD TYPE
1499                              <1>     ;mov    si, ax                  ; SI = DASD TYPE
1500                              <1>     ; 11/04/2021
1501 00004645 89F0                <1>     mov     eax, esi
1502 00004647 0FB6F0              <1>     movzx   esi, al
1503 0000464A 80A7[13770100]0F    <1>     and     byte [DSK_STATE+edi], ~(MED_DET+DBL_STEP+RATE_MSK) ; CLEAR STATE
1504                              <1>     ;dec    si                      ; CHECK FOR 320/360K MEDIA & DRIVE
1505                              <1>     ; 11/04/2021
1506 00004651 4E                  <1>     dec     esi
1507 00004652 7509                <1>     jnz     short NOT_320           ; BYPASS IF NOT
1508 00004654 808F[13770100]90    <1>     or      byte [DSK_STATE+edi], MED_DET+RATE_250 ; SET TO 320/360
1509 0000465B EB45                <1>     jmp     short S0
1510                              <1>
1511                              <1> NOT_320:
1512 0000465D E844020000          <1>     call    MED_CHANGE              ; CHECK FOR TIME_OUT
1513 00004662 803D[08770100]80    <1>     cmp     byte [DSKETTE_STATUS], TIME_OUT
1514 00004669 7437                <1>     jz      short S0                ; IF TIME OUT TELL CALLER
1515                              <1> S3:
1516                              <1>     ;dec    si                      ; CHECK FOR 320/360K IN 1.2M DRIVE
1517                              <1>     ; 11/04/2021
1518 0000466B 4E                  <1>     dec     esi
1519 0000466C 7509                <1>     jnz     short NOT_320_12        ; BYPASS IF NOT
1520 0000466E 808F[13770100]70    <1>     OR      byte [DSK_STATE+edi], MED_DET+DBL_STEP+RATE_300 ; SET STATE
1521 00004675 EB2B                <1>     jmp     short S0
1522                              <1>
1523                              <1> NOT_320_12:
1524                              <1>     ;dec    si                      ; CHECK FOR 1.2M MEDIA IN 1.2M DRIVE
1525                              <1>     ; 11/04/2021
1526 00004677 4E                  <1>     dec     esi
1527 00004678 7509                <1>     jnz     short NOT_12            ; BYPASS IF NOT
1528 0000467A 808F[13770100]10    <1>     or      byte [DSK_STATE+edi], MED_DET+RATE_500 ; SET STATE VARIABLE
1529 00004681 EB1F                <1>     jmp     short S0                ; RETURN TO CALLER
1530                              <1>
1531                              <1> NOT_12:
1532                              <1>     ;dec    si                      ; CHECK FOR SET DASD TYPE 04
1533                              <1>     ; 11/04/2021
1534 00004683 4E                  <1>     dec     esi
1535 00004684 7525                <1>     jnz     short FS_ERR            ; BAD COMMAND EXIT IF NOT VALID TYPE
1536                              <1>
1537 00004686 F687[13770100]04    <1>     test    byte [DSK_STATE+edi], DRV_DET ; DRIVE DETERMINED ?
1538 0000468D 740B                <1>     jz      short ASSUME            ; IF STILL NOT DETERMINED ASSUME
1539 0000468F B050                <1>     mov     al, MED_DET+RATE_300
1540 00004691 F687[13770100]02    <1>         test    byte [DSK_STATE+edi], FMT_CAPA ; MULTIPLE FORMAT CAPABILITY ?
1541 00004698 7502                <1>     jnz     short OR_IT_IN          ; IF 1.2 M THEN DATA RATE 300
1542                              <1>
1543                              <1> ASSUME:
1544 0000469A B090                <1>     mov     al, MED_DET+RATE_250    ; SET UP
1545                              <1>
1546                              <1> OR_IT_IN:
1547 0000469C 0887[13770100]      <1>     or      [DSK_STATE+edi], al     ; OR IN THE CORRECT STATE
1548                              <1> S0:
1549                              <1>     ; 06/08/2022
1550                              <1>     ;call   XLAT_OLD                ; TRANSLATE STATE TO COMPATIBLE MODE
1551 000046A2 E8D5FDFFFF          <1>     call    SETUP_END               ; VARIOUS CLEANUPS
1552                              <1>     ;pop    BX                      ; GET SAVED AL TO BL
1553                              <1>     ; 11/04/2021
1554 000046A7 5B                  <1>     pop     ebx
1555 000046A8 88D8                <1>     mov     al, bl                  ; PUT BACK FOR RETURN
1556 000046AA C3                  <1>     retn
1557                              <1>
1558                              <1> FS_ERR:
1559 000046AB C605[08770100]01    <1>     mov     byte [DSKETTE_STATUS], BAD_CMD ; UNKNOWN STATE,BAD COMMAND
1560 000046B2 EBEE                <1>     jmp     short S0
1561                              <1>
1562                              <1> ;--------------------------------------------------------------------------------
1563                              <1> ; SET_MEDIA (AH = 18H)
1564                              <1> ;   THIS ROUTINE SETS THE TYPE OF MEDIA AND DATA RATE
1565                              <1> ;   TO BE USED FOR THE FOLLOWING FORMAT OPERATION.
1566                              <1> ;
1567                              <1> ; ON ENTRY:
1568                              <1> ;   [BP]   = SECTOR PER TRACK
1569                              <1> ;   [BP+1] = TRACK #
1570                              <1> ;   DI     = DRIVE #
1571                              <1> ;
1572                              <1> ; ON EXIT:
1573                              <1> ;   @DSKETTE_STATUS REFLECTS STATUS
1574                              <1> ;   IF NO ERROR:
1575                              <1> ;           AH = 0
1576                              <1> ;           CY = 0
1577                              <1> ;           ES = SEGMENT OF MEDIA/DRIVE PARAMETER TABLE
1578                              <1> ;           DI/[BP+6] = OFFSET OF MEDIA/DRIVE PARAMETER TABLE
1579                              <1> ;   IF ERROR:
1580                              <1> ;           AH = @DSKETTE_STATUS
1581                              <1> ;           CY = 1
1582                              <1> ;--------------------------------------------------------------------------------
1583                              <1> SET_MEDIA:
1584                              <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
1585 000046B4 E8EC000000          <1>     call    XLAT_NEW                ; TRANSLATE STATE TO PRESENT ARCH.
1586 000046B9 F687[13770100]01    <1>         test    byte [DSK_STATE+edi], TRK_CAPA ; CHECK FOR CHANGE LINE AVAILABLE
1587 000046C0 7415                <1>     jz      short SM_CMOS           ; JUMP IF 40 TRACK DRIVE
1588 000046C2 E8DF010000          <1>     call    MED_CHANGE              ; RESET CHANGE LINE
1589 000046C7 803D[08770100]80    <1>     cmp     byte [DSKETTE_STATUS], TIME_OUT ; IF TIME OUT TELL CALLER
1590 000046CE 746A                <1>     je      short SM_RTN
1591 000046D0 C605[08770100]00    <1>     mov     byte [DSKETTE_STATUS], 0 ; CLEAR STATUS
1592                              <1> SM_CMOS:
1593 000046D7 E827050000          <1>     call    CMOS_TYPE               ; RETURN DRIVE TYPE IN (AL)
1594                              <1>     ;;20/02/2015
1595                              <1>     ;;jc    short MD_NOT_FND        ; ERROR IN CMOS
1596                              <1>     ;;or    al, al                  ; TEST FOR NO DRIVE
1597 000046DC 745C                <1>     jz      short SM_RTN            ; RETURN IF SO
1598 000046DE E85C000000          <1>     call    DR_TYPE_CHECK           ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
1599 000046E3 7233                <1>     jc      short MD_NOT_FND        ; TYPE NOT IN TABLE (BAD CMOS)
1600 000046E5 57                  <1>     push    edi                     ; SAVE REG.
1601 000046E6 31DB                <1>     xor     ebx, ebx                ; BX = INDEX TO DR. TYPE TABLE
```

```
1602 000046E8 B906000000          <1>     mov     ecx, DR_CNT             ; CX = LOOP COUNT
1603                              <1> DR_SEARCH:
1604 000046ED 8AA3[F8650000]      <1>     mov     ah, [DR_TYPE+ebx]       ; GET DRIVE TYPE
1605 000046F3 80E47F              <1>     and     ah, BIT7OFF             ; MASK OUT MSB
1606 000046F6 38E0                <1>     cmp     al, ah                  ; DRIVE TYPE MATCH ?
1607 000046F8 7518                <1>     jne     short NXT_MD            ; NO, CHECK NEXT DRIVE TYPE
1608                              <1> DR_FND:
1609 000046FA 8BBB[F9650000]      <1>     mov     edi, [DR_TYPE+ebx+1]    ; DI = MEDIA/DRIVE PARAM TABLE
1610                              <1> MD_SEARCH:
1611 00004700 8A6704              <1>         mov ah, [edi+MD.SEC_TRK]    ; GET SECTOR/TRACK
1612                              <1>     ;cmp    [ebp], ah               ; MATCH?
1613                              <1>     ; 06/08/2022
1614 00004703 38642418            <1>     cmp     [esp+24], ah  ; CL ; spt
1615 00004707 7509                <1>     jne     short NXT_MD            ; NO, CHECK NEXT MEDIA
1616 00004709 8A670B              <1>         mov     ah, [edi+MD.MAX_TRK]    ; GET MAX. TRACK #
1617                              <1>     ;cmp    [ebp+1], ah             ; MATCH?
1618                              <1>     ; 06/08/2022
1619 0000470C 38642419            <1>     cmp     [esp+25], ah  ; CH ; heads - 1
1620 00004710 740F                <1>     je      short MD_FND            ; YES, GO GET RATE
1621                              <1> NXT_MD:
1622                              <1>     ;add    bx, 3                   ; CHECK NEXT DRIVE TYPE
1623 00004712 83C305              <1>         add ebx, 5 ; 18/02/2015
1624 00004715 E2D6                <1>     loop    DR_SEARCH
1625 00004717 5F                  <1>     pop     edi                     ; RESTORE REG.
1626                              <1> MD_NOT_FND:
1627 00004718 C605[08770100]0C    <1>     mov     byte [DSKETTE_STATUS], MED_NOT_FND ; ERROR, MEDIA TYPE NOT FOUND
1628 0000471F EB19                <1>     jmp     short SM_RTN            ; RETURN
1629                              <1> MD_FND:
1630 00004721 8A470C              <1>         mov al, [edi+MD.RATE]       ; GET RATE
1631 00004724 3C40                <1>     cmp     al, RATE_300            ; DOUBLE STEP REQUIRED FOR RATE 300
1632 00004726 7502                <1>     jne     short MD_SET
1633 00004728 0C20                <1>     or      al, DBL_STEP
1634                              <1> MD_SET:
1635                              <1>     ;;mov   [bp+6], di              ; SAVE TABLE POINTER IN STACK
1636                              <1>     ; 06/08/2022
1637                              <1>     ;mov    [ebp+12], edi ; 18/02/2015
1638                              <1>
1639 0000472A 0C10                <1>     or      al, MED_DET             ; SET MEDIA ESTABLISHED
1640 0000472C 5F                  <1>     pop     edi
1641 0000472D 80A7[13770100]0F    <1>     and     byte [DSK_STATE+edi], ~(MED_DET+DBL_STEP+RATE_MSK) ; CLEAR STATE
1642 00004734 0887[13770100]      <1>     or      [DSK_STATE+edi], al
1643                              <1>     ;mov    ax, cs                  ; SEGMENT OF MEDIA/DRIVE PARAMETER TABLE
1644                              <1>     ;mov    es, ax                  ; ES IS SEGMENT OF TABLE
1645                              <1> SM_RTN:
1646                              <1>     ; 06/08/2022
1647                              <1>     ;call   XLAT_OLD                ; TRANSLATE STATE TO COMPATIBLE MODE
1648                              <1>     ;call   SETUP_END               ; VARIOUS CLEANUPS
1649                              <1>     ;retn
1650                              <1>     ; 06/08/2022
1651 0000473A E93DFDFFFF          <1>     jmp     SETUP_END
1652                              <1>
1653                              <1> ;--------------------------------------------------------------
1654                              <1> ; DR_TYPE_CHECK                                                :
1655                              <1> ;   CHECK IF THE GIVEN DRIVE TYPE IN REGISTER (AL)             :
1656                              <1> ;   IS SUPPORTED IN BIOS DRIVE TYPE TABLE                      :
1657                              <1> ; ON ENTRY:                                          :
1658                              <1> ;   AL = DRIVE TYPE                                           :
1659                              <1> ; ON EXIT:                                           :
1660                              <1> ;   CY = 0  DRIVE TYPE SUPPORTED                     :
1661                              <1> ;        EBX = OFFSET TO MEDIA/DRIVE PARAMETER TABLE   :
1662                              <1> ;   CY = 1  DRIVE TYPE NOT SUPPORTED                 :
1663                              <1> ; REGISTERS ALTERED: EBX, AH ; 11/07/2022                      :
1664                              <1> ;--------------------------------------------------------------
1665                              <1> DR_TYPE_CHECK:
1666                              <1>     ; 09/08/2022 - TRDOS 386 Kernel v2.0.5
1667                              <1>     ; 12/07/2022
1668                              <1>     ; 11/07/2022
1669                              <1>     ; 08/07/2022 - Retro UNIX 386 v1.1 (Kernel v0.2.1.5)
1670                              <1>     ; 24/12/2021
1671                              <1>     ;push   eax ; 11/07/2022
1672                              <1>     ;push   ecx ; 08/07/2022
1673                              <1>     ;xor    ebx,ebx                 ; EBX = INDEX TO DR_TYPE TABLE
1674 0000473F BB[F8650000]        <1>     mov     ebx, DR_TYPE
1675                              <1>     ;;mov   ecx, DR_CNT             ; ECX = LOOP COUNT
1676                              <1>     ;mov    cl, DR_CNT
1677 00004744 B406                <1>     mov     ah, DR_CNT ; 11/07/2022
1678                              <1> TYPE_CHK:
1679                              <1>     ;;mov   ah, [DR_TYPE+ebx]       ; GET DRIVE TYPE
1680                              <1>     ;mov    ah, [ebx]
1681                              <1>     ;cmp    al, ah                  ; DRIVE TYPE MATCH?
1682 00004746 3A03                <1>     cmp     al, [ebx] ; 11/07/2022
1683 00004748 740E                <1>     je      short DR_TYPE_VALID     ; YES, RETURN WITH CARRY RESET
1684                              <1>     ; 16/02/2015 (32 bit address modification)
1685 0000474A 83C305              <1>     add     ebx, 5                  ; CHECK NEXT DRIVE TYPE
1686                              <1>     ;loop   TYPE_CHK
1687                              <1>     ;dec    cl
1688 0000474D FECC                <1>     dec     ah ; 11/07/2022
1689 0000474F 75F5                <1>     jnz     short TYPE_CHK
1690                              <1>     ;
1691 00004751 BB[57660000]        <1>     mov     ebx, MD_TBL6            ; 1.44MB fd parameter table
1692                              <1>                                     ; Default for GET_PARM (11/12/2014)
1693                              <1>     ;
1694 00004756 F9                  <1>     stc                             ; DRIVE TYPE NOT FOUND IN TABLE
1695                              <1>     ;jmp    short TYPE_RTN
1696                              <1>     ; 12/07/2022
1697 00004757 C3                  <1>     retn
1698                              <1> DR_TYPE_VALID:
1699                              <1>     ;mov    ebx, [DR_TYPE+ebx+1]    ; EBX = MEDIA TABLE
1700 00004758 43                  <1>     inc     ebx
1701 00004759 8B1B                <1>     mov     ebx, [ebx]
1702                              <1> TYPE_RTN:
1703                              <1>     ;pop    ecx ; 08/07/2022
1704                              <1>     ; 24/12/2021
1705                              <1>     ;pop    eax ; 11/07/2022
1706 0000475B C3                  <1>     retn
1707                              <1>
1708                              <1> ;--------------------------------------------------------------
1709                              <1> ; SEND_SPEC                                                    :
1710                              <1> ;   SEND THE SPECIFY COMMAND TO CONTROLLER USING DATA FROM     :
1711                              <1> ;   THE DRIVE PARAMETER TABLE POINTED BY @DISK_POINTER  :
1712                              <1> ; ON ENTRY: @DISK_POINTER = DRIVE PARAMETER TABLE              :
1713                              <1> ; ON EXIT:  NONE                                     :
1714                              <1> ; REGISTERS ALTERED: CX, DX                                    :
1715                              <1> ;--------------------------------------------------------------
1716                              <1> SEND_SPEC:
1717                              <1>     ; 06/08/2022
1718 0000475C 50                  <1>     push    eax                     ; SAVE AX
1719 0000475D B8[83470000]        <1>     mov     eax, SPECBAC            ; LOAD ERROR ADDRESS
1720 00004762 50                  <1>     push    eax                     ; PUSH NEC_OUT ERROR RETURN
1721 00004763 B403                <1>     mov     ah, 03h                 ; SPECIFY COMMAND
1722 00004765 E898050000          <1>     call    NEC_OUTPUT              ; OUTPUT THE COMMAND
1723                              <1>     ;sub    dl, dl                  ; FIRST SPECIFY BYTE
1724                              <1>     ; 06/08/2022
1725 0000476A 28C0                <1>     sub     al, al ; 0
```

```
1726 0000476C E89B040000          <1>     call    GET_PARM                ; GET PARAMETER TO AH
1727 00004771 E88C050000          <1>     call    NEC_OUTPUT              ; OUTPUT THE COMMAND
1728                              <1>     ;mov    dl, 1                   ; SECOND SPECIFY BYTE
1729                              <1>     ; 06/08/2022
1730 00004776 B001                <1>     mov     al, 1
1731 00004778 E88F040000          <1>     call    GET_PARM                ; GET PARAMETER TO AH
1732 0000477D E880050000          <1>     call    NEC_OUTPUT              ; OUTPUT THE COMMAND
1733 00004782 58                  <1>     pop     eax                     ; POP ERROR RETURN
1734                              <1> SPECBAC:
1735 00004783 58                  <1>     pop     eax                     ; RESTORE ORIGINAL AX VALUE
1736 00004784 C3                  <1>     retn
1737                              <1>
1738                              <1> ;---------------------------------------------------------------
1739                              <1> ; SEND_SPEC_MD                                                 :
1740                              <1> ;   SEND THE SPECIFY COMMAND TO CONTROLLER USING DATA FROM     :
1741                              <1> ;   THE MEDIA/DRIVE PARAMETER TABLE POINTED BY (CS:BX)  :
1742                              <1> ; ON ENTRY: CS:BX = MEDIA/DRIVE PARAMETER TABLE          :
1743                              <1> ; ON EXIT:  NONE                                         :
1744                              <1> ; REGISTERS ALTERED: AX                                        :
1745                              <1> ;---------------------------------------------------------------
1746                              <1> SEND_SPEC_MD:
1747 00004785 50                  <1>     push    eax                     ; SAVE RATE DATA
1748 00004786 B8[A3470000]        <1>     mov     eax, SPEC_ESBAC               ; LOAD ERROR ADDRESS
1749 0000478B 50                  <1>     push    eax                     ; PUSH NEC_OUT ERROR RETURN
1750 0000478C B403                <1>     mov     ah, 03h                 ; SPECIFY COMMAND
1751 0000478E E86F050000          <1>     call    NEC_OUTPUT              ; OUTPUT THE COMMAND
1752 00004793 8A23                <1>         mov     ah, [ebx+MD.SPEC1]      ; GET 1ST SPECIFY BYTE
1753 00004795 E868050000          <1>     call    NEC_OUTPUT              ; OUTPUT THE COMMAND
1754 0000479A 8A6301              <1>         mov     ah, [ebx+MD.SPEC2]      ; GET SECOND SPECIFY BYTE
1755 0000479D E860050000          <1>     call    NEC_OUTPUT              ; OUTPUT THE COMMAND
1756 000047A2 58                  <1>     pop     eax                     ; POP ERROR RETURN
1757                              <1> SPEC_ESBAC:
1758 000047A3 58                  <1>     pop     eax                     ; RESTORE ORIGINAL AX VALUE
1759 000047A4 C3                  <1>     retn
1760                              <1>
1761                              <1> ;-----------------------------------------------------------------------------
1762                              <1> ; XLAT_NEW
1763                              <1> ;   TRANSLATES DISKETTE STATE LOCATIONS FROM COMPATIBLE
1764                              <1> ;   MODE TO NEW ARCHITECTURE.
1765                              <1> ;
1766                              <1> ; ON ENTRY: DI = DRIVE #
1767                              <1> ;-----------------------------------------------------------------------------
1768                              <1> XLAT_NEW:
1769                              <1>     ; 06/08/2022 - TRDOS 386 Kernel v2.0.5
1770                              <1>     ; 11/07/2022 - Retro UNIX 386 v1.1 (Kernel v0.2.1.5)
1771 000047A5 83FF01              <1>     cmp     edi, 1                  ; VALID DRIVE
1772 000047A8 7709                <1>     ja      short XN_OUT            ; IF INVALID BACK
1773 000047AA 80BF[13770100]00    <1>     cmp     byte [DSK_STATE+edi], 0       ; NO DRIVE ?
1774 000047B1 7401                <1>     jz      short DO_DET            ; IF NO DRIVE ATTEMPT DETERMINE
1775                              <1>
1776                              <1>     ;;mov   cx, di                  ; CX = DRIVE NUMBER
1777                              <1>     ;mov    ecx, edi
1778                              <1>     ;or     cl, cl
1779                              <1>     ;jz     short XN_0
1780                              <1>     ;shl    cl, 2                   ; CL = SHIFT COUNT, A=0, B=4
1781                              <1>     ;mov    al, [HF_CNTRL]          ; DRIVE INFORMATION
1782                              <1>     ;ror    al, cl                  ; TO LOW NIBBLE
1783                              <1> ;XN_0:
1784                              <1>     ;and    al, DRV_DET+FMT_CAPA+TRK_CAPA ; KEEP DRIVE BITS
1785                              <1>         ;and       byte [DSK_STATE+edi], ~(DRV_DET+FMT_CAPA+TRK_CAPA)
1786                              <1>     ;or     [DSK_STATE+edi], al     ; UPDATE DRIVE STATE
1787                              <1> XN_OUT:
1788 000047B3 C3                  <1>     retn
1789                              <1>
1790                              <1> DO_DET:
1791                              <1>     ;call   DRIVE_DET               ; TRY TO DETERMINE
1792                              <1>     ;retn
1793                              <1>     ;jmp    DRIVE_DET
1794                              <1>
1795                              <1> ;-----------------------------------------------------------------------------
1796                              <1> ; DRIVE_DET
1797                              <1> ;   DETERMINES WHETHER DRIVE IS 80 OR 40 TRACKS AND
1798                              <1> ;   UPDATES STATE INFORMATION ACCORDINGLY.
1799                              <1> ; ON ENTRY: DI = DRIVE #
1800                              <1> ;-----------------------------------------------------------------------------
1801                              <1> DRIVE_DET:
1802                              <1>     ; 06/08/2022 - TRDOS 386 Kernel v2.0.5
1803                              <1>     ; 08/07/2022 - Retro UNIX 386 v1.1 (Kernel v0.2.1.5)
1804 000047B4 E88E040000          <1>     call    MOTOR_ON                ; TURN ON MOTOR IF NOT ALREADY ON
1805 000047B9 E8FA050000          <1>     call    RECAL                   ; RECALIBRATE DRIVE
1806 000047BE 724E                <1>     jc      short DD_BAC            ; ASSUME NO DRIVE PRESENT
1807 000047C0 B530                <1>     mov     ch, TRK_SLAP            ; SEEK TO TRACK 48
1808 000047C2 E872050000          <1>     call    SEEK
1809 000047C7 7245                <1>     jc      short DD_BAC            ; ERROR NO DRIVE
1810 000047C9 B50B                <1>     mov     ch, QUIET_SEEK+1        ; SEEK TO TRACK 10
1811                              <1> SK_GIN:
1812 000047CB FECD                <1>     dec     ch                      ; DECREMENT TO NEXT TRACK
1813                              <1>     ;push   cx                      ; SAVE TRACK
1814                              <1>     ; 11/04/2021
1815 000047CD 51                  <1>     push    ecx
1816 000047CE E866050000          <1>     call    SEEK
1817 000047D3 723A                <1>     jc      short POP_BAC           ; POP AND RETURN
1818 000047D5 B8[0F480000]        <1>     mov     eax, POP_BAC            ; LOAD NEC OUTPUT ERROR ADDRESS
1819 000047DA 50                  <1>     push    eax
1820 000047DB B404                <1>     mov     ah, SENSE_DRV_ST        ; SENSE DRIVE STATUS COMMAND BYTE
1821 000047DD E820050000          <1>     call    NEC_OUTPUT              ; OUTPUT TO NEC
1822                              <1>     ;mov    ax, di                  ; AL = DRIVE
1823                              <1>     ; 06/08/2022
1824 000047E2 89F8                <1>     mov     eax, edi
1825 000047E4 88C4                <1>     mov     ah, al                  ; AH = DRIVE
1826 000047E6 E817050000          <1>     call    NEC_OUTPUT              ; OUTPUT TO NEC
1827 000047EB E849060000          <1>     call    RESULTS                 ; GO GET STATUS
1828 000047F0 58                  <1>     pop     eax                     ; THROW AWAY ERROR ADDRESS
1829                              <1>     ;pop    cx                      ; RESTORE TRACK
1830                              <1>     ; 11/04/2021
1831 000047F1 59                  <1>     pop     ecx
1832 000047F2 F605[09770100]10    <1>     test    byte [NEC_STATUS], HOME       ; TRACK 0 ?
1833 000047F9 74D0                <1>     jz      short SK_GIN            ; GO TILL TRACK 0
1834 000047FB 08ED                <1>     or      ch, ch                  ; IS HOME AT TRACK 0
1835 000047FD 7408                <1>     jz      short IS_80             ; MUST BE 80 TRACK DRIVE
1836                              <1>
1837                              <1> ;   DRIVE IS A 360; SET DRIVE TO DETERMINED;
1838                              <1> ;   SET MEDIA TO DETERMINED AT RATE 250.
1839                              <1>
1840 000047FF 808F[13770100]94    <1>     or      byte [DSK_STATE+edi], DRV_DET+MED_DET+RATE_250
1841 00004806 C3                  <1>     retn                            ; ALL INFORMATION SET
1842                              <1> IS_80:
1843 00004807 808F[13770100]01    <1>     or      byte [DSK_STATE+edi], TRK_CAPA ; SETUP 80 TRACK CAPABILITY
1844                              <1> DD_BAC:
1845 0000480E C3                  <1>     retn
1846                              <1> POP_BAC:
1847                              <1>     ;pop    cx                      ; THROW AWAY
1848                              <1>     ; 11/04/2021
1849 0000480F 59                  <1>     pop     ecx
```

```
1850 00004810 C3                <1>     retn
1851                            <1>
1852                            <1>
1853                            <1>     ; 06/08/2022 - TRDOS 386 Kernel v2.0.5
1854                            <1>
1855                            <1> %if 0
1856                            <1>
1857                            <1> ;--------------------------------------------------------------------------------
1858                            <1> ; XLAT_OLD
1859                            <1> ;   TRANSLATES DISKETTE STATE LOCATIONS FROM NEW
1860                            <1> ;   ARCHITECTURE TO COMPATIBLE MODE.
1861                            <1> ;
1862                            <1> ; ON ENTRY: DI = DRIVE
1863                            <1> ;--------------------------------------------------------------------------------
1864                            <1> XLAT_OLD:
1865                            <1>     cmp     edi, 1                  ; VALID DRIVE ?
1866                            <1>     ja      short XO_OUT              ; IF INVALID BACK
1867                            <1>         cmp byte [DSK_STATE+edi], 0       ; NO DRIVE ?
1868                            <1>     jz      short XO_OUT            ; IF NO DRIVE TRANSLATE DONE
1869                            <1>
1870                            <1> ;-----      TEST FOR SAVED DRIVE INFORMATION ALREADY SET
1871                            <1>
1872                            <1>     ;mov    cx, di                  ; CX = DRIVE NUMBER
1873                            <1>     ; 06/08/2022
1874                            <1>     mov     ecx, edi
1875                            <1>     shl     cl, 2                   ; CL = SHIFT COUNT, A=0, B=4
1876                            <1>     mov     ah, FMT_CAPA            ; LOAD MULTIPLE DATA RATE BIT MASK
1877                            <1>     ror     ah, cl                  ; ROTATE BY MASK
1878                            <1>     test    [HF_CNTRL], ah          ; MULTIPLE-DATA RATE DETERMINED ?
1879                            <1>     jnz     short SAVE_SET          ; IF SO, NO NEED TO RE-SAVE
1880                            <1>
1881                            <1> ;-----      ERASE DRIVE BITS IN @HF_CNTRL FOR THIS DRIVE
1882                            <1>
1883                            <1>     mov     ah, DRV_DET+FMT_CAPA+TRK_CAPA ; MASK TO KEEP
1884                            <1>     ror     ah, cl                  ; FIX MASK TO KEEP
1885                            <1>     not     ah                      ; TRANSLATE MASK
1886                            <1>     and     [HF_CNTRL], ah          ; KEEP BITS FROM OTHER DRIVE INTACT
1887                            <1>
1888                            <1> ;-----      ACCESS CURRENT DRIVE BITS AND STORE IN @HF_CNTRL
1889                            <1>
1890                            <1>     mov     al, [DSK_STATE+edi]     ; ACCESS STATE
1891                            <1>     and     al, DRV_DET+FMT_CAPA+TRK_CAPA ; KEEP DRIVE BITS
1892                            <1>     ror     al, cl                  ; FIX FOR THIS DRIVE
1893                            <1>     or      [HF_CNTRL], al          ; UPDATE SAVED DRIVE STATE
1894                            <1>
1895                            <1> ;-----      TRANSLATE TO COMPATIBILITY MODE
1896                            <1>
1897                            <1> SAVE_SET:
1898                            <1>     mov     ah, [DSK_STATE+edi]     ; ACCESS STATE
1899                            <1>     mov     bh, ah                  ; TO BH FOR LATER
1900                            <1>     and     ah, RATE_MSK            ; KEEP ONLY RATE
1901                            <1>     cmp     ah, RATE_500            ; RATE 500 ?
1902                            <1>     jz      short CHK_144           ; YES 1.2/1.2 OR 1.44/1.44
1903                            <1>     mov     al, M3D1U               ; AL = 360 IN 1.2 UNESTABLISHED
1904                            <1>     cmp     ah, RATE_300            ; RATE 300 ?
1905                            <1>     jnz     short CHK_250           ; NO, 360/360, 720/720 OR 720/1.44
1906                            <1>     test    bh, DBL_STEP            ; CHECK FOR DOUBLE STEP
1907                            <1>     jnz     short TST_DET           ; MUST BE 360 IN 1.2
1908                            <1> UNKNO:
1909                            <1>     mov     al, MED_UNK             ; NONE OF THE ABOVE
1910                            <1>     jmp     short AL_SET            ; PROCESS COMPLETE
1911                            <1> CHK_144:
1912                            <1>     call    CMOS_TYPE               ; RETURN DRIVE TYPE IN (AL)
1913                            <1>     ;;20/02/2015
1914                            <1>     ;;jc    short UNKNO             ; ERROR, SET 'NONE OF ABOVE'
1915                            <1>     jz      short UNKNO ;; 20/02/2015
1916                            <1>     cmp     al, 2                   ; 1.2MB DRIVE ?
1917                            <1>     jne     short UNKNO             ; NO, GO SET 'NONE OF ABOVE'
1918                            <1>     mov     al, M1D1U               ; AL = 1.2 IN 1.2 UNESTABLISHED
1919                            <1>     jmp     short TST_DET
1920                            <1> CHK_250:
1921                            <1>     mov     al, M3D3U               ; AL = 360 IN 360 UNESTABLISHED
1922                            <1>     cmp     ah, RATE_250            ; RATE 250 ?
1923                            <1>     jnz     short UNKNO             ; IF SO FALL IHRU
1924                            <1>     test    bh, TRK_CAPA            ; 80 TRACK CAPABILITY ?
1925                            <1>     jnz     short UNKNO             ; IF SO JUMP, FALL THRU TEST DET
1926                            <1> TST_DET:
1927                            <1>     test    bh, MED_DET             ; DETERMINED ?
1928                            <1>     jz      short AL_SET            ; IF NOT THEN SET
1929                            <1>     add     al, 3                   ; MAKE DETERMINED/ESTABLISHED
1930                            <1> AL_SET:
1931                            <1>     and     byte [DSK_STATE+edi], ~(DRV_DET+FMT_CAPA+TRK_CAPA) ; CLEAR DRIVE
1932                            <1>     or      [DSK_STATE+edi], al     ; REPLACE WITH COMPATIBLE MODE
1933                            <1> XO_OUT:
1934                            <1>     retn
1935                            <1>
1936                            <1> %endif
1937                            <1>
1938                            <1> ;--------------------------------------------------------------------------------
1939                            <1> ; SETUP_STATE:     INITIALIZES START AND END RATES.
1940                            <1> ;--------------------------------------------------------------------------------
1941                            <1> SETUP_STATE:
1942 00004811 F687[13770100]10 <1>     test    byte [DSK_STATE+edi], MED_DET ; MEDIA DETERMINED ?
1943 00004818 7537             <1>     jnz     short J1C               ; NO STATES IF DETERMINED
1944 0000481A 66B84000         <1>         mov     ax, (RATE_500*256)+RATE_300 ; AH = START RATE, AL = END RATE
1945 0000481E F687[13770100]04 <1>     test    byte [DSK_STATE+edi], DRV_DET ; DRIVE ?
1946 00004825 740D             <1>     jz      short AX_SET            ; DO NOT KNOW DRIVE
1947 00004827 F687[13770100]02 <1>     test    byte [DSK_STATE+edi], FMT_CAPA ; MULTI-RATE?
1948 0000482E 7504             <1>     jnz     short AX_SET            ; JUMP IF YES
1949 00004830 66B88080         <1>         mov     ax, RATE_250*257  ; START A END RATE 250 FOR 360 DRIVE
1950                            <1> AX_SET:
1951 00004834 80A7[13770100]1F <1>     and     byte [DSK_STATE+edi], ~(RATE_MSK+DBL_STEP) ; TURN OFF THE RATE
1952 0000483B 08A7[13770100]   <1>     or      [DSK_STATE+edi], ah     ; RATE FIRST TO TRY
1953 00004841 8025[10770100]F3 <1>     and     byte [LASTRATE], ~STRT_MSK ; ERASE LAST TO TRY RATE BITS
1954 00004848 C0C804           <1>     ror     al, 4                   ; TO OPERATION LAST RATE LOCATION
1955 0000484B 0805[10770100]   <1>     or      [LASTRATE], al          ; LAST RATE
1956                            <1> J1C:
1957 00004851 C3               <1>     retn
1958                            <1>
1959                            <1> ;--------------------------------------------------------------------------------
1960                            <1> ;  FMT_INIT: ESTABLISH STATE IF UNESTABLISHED AT FORMAT TIME.
1961                            <1> ;--------------------------------------------------------------------------------
1962                            <1> FMT_INIT:
1963 00004852 F687[13770100]10 <1>     test    byte [DSK_STATE+edi], MED_DET ; IS MEDIA ESTABLISHED
1964 00004859 7546             <1>     jnz     short F1_OUT            ; IF SO RETURN
1965 0000485B E8A3030000       <1>     call    CMOS_TYPE               ; RETURN DRIVE TYPE IN AL
1966                            <1>     ;; 20/02/2015
1967                            <1>     ;;jc    short CL_DRV            ; ERROR IN CMOS ASSUME NO DRIVE
1968 00004860 7440             <1>     jz      short CL_DRV ;; 20/02/2015
1969 00004862 FEC8             <1>     dec     al                      ; MAKE ZERO ORIGIN
1970                            <1>     ;;JS    short CL_DRV            ; NO DRIVE IF AL 0
1971 00004864 8AA7[13770100]   <1>     mov     ah, [DSK_STATE+edi]     ; AH = CURRENT STATE
1972 0000486A 80E40F           <1>     and     ah, ~(MED_DET+DBL_STEP+RATE_MSK) ; CLEAR
1973 0000486D 08C0             <1>     or      al, al                  ; CHECK FOR 360
```

```
1974 0000486F 7505                <1>     jnz     short N_360             ; IF 360 WILL BE 0
1975 00004871 80CC90              <1>     or      ah, MED_DET+RATE_250    ; ESTABLISH MEDIA
1976 00004874 EB25                <1>     jmp     short SKP_STATE             ; SKIP OTHER STATE PROCESSING
1977                              <1> N_360:
1978 00004876 FEC8                <1>     dec     al                      ; 1.2 M DRIVE
1979 00004878 7505                <1>     jnz     short N_12              ; JUMP IF NOT
1980                              <1> F1_RATE:
1981 0000487A 80CC10              <1>     or      ah, MED_DET+RATE_500    ; SET FORMAT RATE
1982 0000487D EB1C                <1>     jmp     short SKP_STATE             ; SKIP OTHER STATE PROCESSING
1983                              <1> N_12:
1984 0000487F FEC8                <1>     dec     al                      ; CHECK FOR TYPE 3
1985 00004881 750F                <1>     jnz     short N_720             ; JUMP IF NOT
1986 00004883 F6C404              <1>     test    ah, DRV_DET             ; IS DRIVE DETERMINED
1987 00004886 7410                <1>     jz      short ISNT_12           ; TREAT AS NON 1.2 DRIVE
1988 00004888 F6C402              <1>     test    ah, FMT_CAPA            ; IS 1.2M
1989 0000488B 740B                <1>     jz      short ISNT_12           ; JUMP IF NOT
1990 0000488D 80CC50              <1>     or      ah, MED_DET+RATE_300    ; RATE 300
1991 00004890 EB09                <1>     jmp     short SKP_STATE             ; CONTINUE
1992                              <1> N_720:
1993 00004892 FEC8                <1>     dec     al                      ; CHECK FOR TYPE 4
1994 00004894 750C                <1>     jnz     short CL_DRV            ; NO DRIVE, CMOS BAD
1995 00004896 EBE2                <1>     jmp     SHORT F1_RATE
1996                              <1> ISNT_12:
1997 00004898 80CC90              <1>     or      ah, MED_DET+RATE_250    ; MUST BE RATE 250
1998                              <1> SKP_STATE:
1999 0000489B 88A7[13770100]      <1>     mov     [DSK_STATE+edi], ah     ; STORE AWAY
2000                              <1> F1_OUT:
2001 000048A1 C3                  <1>     retn
2002                              <1> CL_DRV:
2003 000048A2 30E4                <1>     xor     ah, ah                  ; CLEAR STATE
2004 000048A4 EBF5                <1>     jmp     short SKP_STATE             ; SAVE IT
2005                              <1>
2006                              <1> ;--------------------------------------------------------------------------------
2007                              <1> ; MED_CHANGE
2008                              <1> ;   CHECKS FOR MEDIA CHANGE, RESETS MEDIA CHANGE,
2009                              <1> ;   CHECKS MEDIA CHANGE AGAIN.
2010                              <1> ;
2011                              <1> ; ON EXIT:  CY = 1 MEANS MEDIA CHANGE OR TIMEOUT
2012                              <1> ;           @DSKETTE_STATUS = ERROR CODE
2013                              <1> ;--------------------------------------------------------------------------------
2014                              <1> MED_CHANGE:
2015                              <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
2016 000048A6 E8E8050000          <1>     call    READ_DSKCHNG            ; READ DISK CHANCE LINE STATE
2017 000048AB 7445                <1>     jz      short MC_OUT            ; BYPASS HANDLING DISK CHANGE LINE
2018 000048AD 80A7[13770100]EF    <1>     and     byte [DSK_STATE+edi], ~MED_DET ; CLEAR STATE FOR THIS DRIVE
2019                              <1>
2020                              <1> ;   THIS SEQUENCE ENSURES WHENEVER A DISKETTE IS CHANGED THAT
2021                              <1> ;   ON THE NEXT OPERATION THE REQUIRED MOTOR START UP TIME WILL
2022                              <1> ;   BE WAITED. (DRIVE MOTOR MAY GO OFF UPON DOOR OPENING).
2023                              <1>
2024                              <1>     ;mov    cx, di                  ; CL = DRIVE 0
2025                              <1>     ; 06/08/2022
2026 000048B4 89F9                <1>     mov     ecx, edi
2027 000048B6 B001                <1>     mov     al, 1                   ; MOTOR ON BIT MASK
2028 000048B8 D2E0                <1>     shl     al, cl                  ; TO APPROPRIATE POSITION
2029 000048BA F6D0                <1>     not     al                      ; KEEP ALL BUT MOTOR ON
2030 000048BC FA                  <1>     cli                             ; NO INTERRUPTS
2031 000048BD 2005[06770100]      <1>     and     [MOTOR_STATUS], al      ; TURN MOTOR OFF INDICATOR
2032 000048C3 FB                  <1>     sti                             ; INTERRUPTS ENABLED
2033 000048C4 E87E030000          <1>     call    MOTOR_ON                ; TURN MOTOR ON
2034                              <1>
2035                              <1> ;-----      THIS SEQUENCE OF SEEKS IS USED TO RESET DISKETTE CHANGE SIGNAL
2036                              <1>
2037 000048C9 E82FFAFFFF          <1>     call    DSK_RESET               ; RESET NEC
2038 000048CE B501                <1>     mov     ch, 1                   ; MOVE TO CYLINDER 1
2039 000048D0 E864040000          <1>     call    SEEK                    ; ISSUE SEEK
2040 000048D5 30ED                <1>     xor     ch, ch                  ; MOVE TO CYLINDER 0
2041 000048D7 E85D040000          <1>     call    SEEK                    ; ISSUE SEEK
2042 000048DC C605[08770100]06    <1>     mov     byte [DSKETTE_STATUS], MEDIA_CHANGE ; STORE IN STATUS
2043                              <1> OK1:
2044 000048E3 E8AB050000          <1>     call    READ_DSKCHNG            ; CHECK MEDIA CHANGED AGAIN
2045 000048E8 7407                <1>     jz      short OK2               ; IF ACTIVE, NO DISKETTE, TIMEOUT
2046                              <1> OK4:
2047 000048EA C605[08770100]80    <1>     mov     byte [DSKETTE_STATUS], TIME_OUT ; TIMEOUT IF DRIVE EMPTY
2048                              <1> OK2:
2049 000048F1 F9                  <1>     stc                             ; MEDIA CHANGED, SET CY
2050                              <1> MC_OUT:     ; 06/08/2022 (cf = 0)
2051 000048F2 C3                  <1>     retn
2052                              <1> ;MC_OUT:
2053                              <1>     ;clc                            ; NO MEDIA CHANGED, CLEAR CY
2054                              <1>     ;retn
2055                              <1>
2056                              <1> ;--------------------------------------------------------------------------------
2057                              <1> ; SEND_RATE
2058                              <1> ;   SENDS DATA RATE COMMAND TO NEC
2059                              <1> ; ON ENTRY: DI = DRIVE #
2060                              <1> ; ON EXIT:  NONE
2061                              <1> ; REGISTERS ALTERED: DX
2062                              <1> ;--------------------------------------------------------------------------------
2063                              <1> SEND_RATE:
2064                              <1>     ;push   ax                      ; SAVE REG.
2065                              <1>     ; 11/04/2021
2066 000048F3 50                  <1>     push    eax
2067 000048F4 8025[10770100]3F    <1>     and     byte [LASTRATE], ~SEND_MSK ; ELSE CLEAR LAST RATE ATTEMPTED
2068 000048FB 8A87[13770100]      <1>     mov     al, [DSK_STATE+edi]     ; GET RATE STATE OF THIS DRIVE
2069 00004901 24C0                <1>     and     al, SEND_MSK            ; KEEP ONLY RATE BITS
2070 00004903 0805[10770100]      <1>     or      [LASTRATE], al          ; SAVE NEW RATE FOR NEXT CHECK
2071 00004909 C0C002              <1>     rol     al, 2                   ; MOVE TO BIT OUTPUT POSITIONS
2072 0000490C 66BAF703            <1>     mov     dx, 03F7h               ; OUTPUT NEW DATA RATE
2073 00004910 EE                  <1>     out     dx, al
2074                              <1>     ;pop    ax                      ; RESTORE REG.
2075                              <1>     ; 11/04/2021
2076 00004911 58                  <1>     pop     eax
2077 00004912 C3                  <1>     retn
2078                              <1>
2079                              <1> ;--------------------------------------------------------------------------------
2080                              <1> ; CHK_LASTRATE
2081                              <1> ;   CHECK PREVIOUS DATE RATE SNT TO THE CONTROLLER.
2082                              <1> ; ON ENTRY:
2083                              <1> ;   DI = DRIVE #
2084                              <1> ; ON EXIT:
2085                              <1> ;   ZF =  1 DATA RATE IS THE SAME AS THE LAST RATE SENT TO NEC
2086                              <1> ;   ZF =  0 DATA RATE IS DIFFERENT FROM LAST RATE
2087                              <1> ; REGISTERS ALTERED: DX
2088                              <1> ;--------------------------------------------------------------------------------
2089                              <1> CHK_LASTRATE:
2090                              <1>     ; 13/07/2022 - TRDOS 386 v2.0.5
2091                              <1>     ;push   ax                      ; SAVE REG.
2092                              <1>     ; 11/04/2021
2093 00004913 50                  <1>     push    eax
2094                              <1>     ; 13/07/2022 (BugFix)
2095 00004914 8A25[10770100]      <1>     mov     ah, [LASTRATE]              ; GET LAST DATA RATE SELECTED
2096 0000491A 8A87[13770100]      <1>     mov     al, [DSK_STATE+edi]     ; GET RATE STATE OF THIS DRIVE
2097 00004920 6625C0C0            <1>         and ax, SEND_MSK*257        ; KEEP ONLY RATE BITS OF BOTH
```

```
                                <1>     cmp     al, ah                  ; COMPARE TO PREVIOUSLY TRIED
                                <1>                                     ; ZF = 1 RATE IS THE SAME
                                <1>     ;pop    ax                      ; RESTORE REG.
                                <1>     ; 11/04/2021
00004926 58                     <1>     pop     eax
00004927 C3                     <1>     retn
                                <1>
                                <1> ;---------------------------------------------------------------------------
                                <1> ; DMA_SETUP
                                <1> ;   THIS ROUTINE SETS UP THE DMA FOR READ/WRITE/VERIFY OPERATIONS.
                                <1> ;
                                <1> ; ON ENTRY: AL = DMA COMMAND
                                <1> ;
                                <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
                                <1> ;---------------------------------------------------------------------------
                                <1>
                                <1> ; SI = Head #, # of Sectors or DASD Type
                                <1>
                                <1> ; 22/08/2015
                                <1> ; 08/02/2015 - Protected Mode Modification
                                <1> ; 06/02/2015 - 07/02/2015
                                <1> ; NOTE: Buffer address must be in 1st 16MB of Physical Memory (24 bit limit).
                                <1> ; (DMA Addres = Physical Address)
                                <1> ; (Retro UNIX 386 v1 Kernel/System Mode Virtual Address = Physical Address)
                                <1> ;
                                <1> ; 09/08/2022
                                <1> ; 06/08/2022
                                <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>     ; 09/08/2022
                                <1>     ; 06/08/2022 - TRDOS 386 Kernel v2.0.5
                                <1>     ;; 20/02/2015
                                <1>     ;;mov   edx, [ebp+4]            ; Buffer address
                                <1>     ; 06/08/2022
                                <1>     ;mov    edx, ebp ; buffer address
                                <1>     ;test   edx, 0FF000000h                 ; 16 MB limit (22/08/2015, bugfix)
00004928 F7C5000000FF           <1>     test    ebp, 0FF000000h
0000492E 7566                   <1>     jnz     short dma_bnd_err_stc
                                <1>     ; 09/08/2022
00004930 89EA                   <1>     mov     edx, ebp
                                <1>     ;
                                <1>     ;;push  ax                      ; DMA command
                                <1>     ; 11/04/2021
00004932 50                     <1>     push    eax
                                <1>     ; 06/08/2022
                                <1>     ;push   edx                     ; *
                                <1>     ;mov    dl, 3                   ; GET BYTES/SECTOR PARAMETER
                                <1>     ; 06/08/2022
00004933 B003                   <1>     mov     al, 3                   ; GET BYTES/SECTOR PARAMETER
00004935 E8D2020000             <1>     call    GET_PARM                ;
0000493A 88E1                   <1>     mov     cl, ah                  ; SHIFT COUNT (0=128, 1=256, 2=512 ETC)
                                <1>     ;mov    ax, si                  ; Sector count
                                <1>     ; 06/08/2022
0000493C 89F0                   <1>     mov     eax, esi
0000493E 88C4                   <1>     mov     ah, al                  ; AH = # OF SECTORS
00004940 28C0                   <1>     sub     al, al                  ; AL = 0, AX = # SECTORS * 256
                                <1>     ;shr    ax, 1                   ; AX = # SECTORS * 128
                                <1>     ; 06/08/2022
00004942 D1E8                   <1>     shr     eax, 1
                                <1>     ;shl    ax, cl                  ; SHIFT BY PARAMETER VALUE
                                <1>     ;dec    ax                      ; -1 FOR DMA VALUE
                                <1>     ;mov    cx, ax
00004944 D3E0                   <1>     shl     eax, cl
00004946 48                     <1>     dec     eax
00004947 89C1                   <1>     mov     ecx, eax
                                <1>     ; 06/08/2022
                                <1>     ;pop    edx                     ; *
                                <1>     ;;pop   ax
                                <1>     ; 11/04/2021
00004949 58                     <1>     pop     eax
0000494A 3C42                   <1>     cmp     al, 42h
0000494C 7507                   <1>         jne     short NOT_VERF
                                <1>     ; 09/08/2022
0000494E BA0000FF00             <1>     mov     edx, 0FF0000h
                                <1>     ; 06/08/2022
                                <1>     ;mov    ebp, 0FF0000h
00004953 EB08                   <1>     jmp     short J33
                                <1> NOT_VERF:
00004955 6601CA                 <1>     add     dx, cx                  ; check for overflow
00004958 723D                   <1>     jc      short dma_bnd_err
                                <1>     ;
0000495A 6629CA                 <1>     sub     dx, cx                  ; Restore start address
                                <1> J33:
0000495D FA                     <1>     cli                             ; DISABLE INTERRUPTS DURING DMA SET-UP
0000495E E60C                   <1>     out     DMA+12, al              ; SET THE FIRST/LA5T F/F
                                <1>     IODELAY                         ; WAIT FOR I/O
  78 00004960 EB00              <2>  jmp short $+2
  79 00004962 EB00              <2>  jmp short $+2
00004964 E60B                   <1>     out     DMA+11, al              ; OUTPUT THE MODE BYTE
                                <1>     ;mov    eax, edx                ; Buffer address
                                <1>     ; 06/08/2022
                                <1>     ;mov    eax, ebp
                                <1>     ; 09/08/2022
00004966 89D0                   <1>     mov     eax, edx
00004968 E604                   <1>     out     DMA+4, al               ; OUTPUT LOW ADDRESS
                                <1>     IODELAY                         ; WAIT FOR I/O
  78 0000496A EB00              <2>  jmp short $+2
  79 0000496C EB00              <2>  jmp short $+2
0000496E 88E0                   <1>     mov     al, ah
00004970 E604                   <1>     out     DMA+4, al               ; OUTPUT HIGH ADDRESS
00004972 C1E810                 <1>     shr     eax, 16
                                <1>     IODELAY                         ; I/O WAIT STATE
  78 00004975 EB00              <2>  jmp short $+2
  79 00004977 EB00              <2>  jmp short $+2
00004979 E681                   <1>     out     081h, al                ; OUTPUT HIGHEST BITS TO PAGE REGISTER
                                <1>     IODELAY
  78 0000497B EB00              <2>  jmp short $+2
  79 0000497D EB00              <2>  jmp short $+2
                                <1>     ;mov    ax, cx                  ; Byte count - 1
                                <1>     ; 06/08/2022
0000497F 89C8                   <1>     mov     eax, ecx
00004981 E605                   <1>     out     DMA+5, al               ; LOW BYTE OF COUNT
                                <1>     IODELAY                         ; WAIT FOR I/O
  78 00004983 EB00              <2>  jmp short $+2
  79 00004985 EB00              <2>  jmp short $+2
00004987 88E0                   <1>     mov     al, ah
00004989 E605                   <1>     out     DMA+5, al               ; HIGH BYTE OF COUNT
                                <1>     IODELAY
  78 0000498B EB00              <2>  jmp short $+2
  79 0000498D EB00              <2>  jmp short $+2
0000498F FB                     <1>     sti                             ; RE-ENABLE INTERRUPTS
```

```
                        <1>     mov     al, 2                   ; MODE FOR 8237
00004992 E60A           <1>     out     DMA+10, al              ; INITIALIZE THE DISKETTE CHANNEL
                        <1>
00004994 F8             <1>     clc     ; 04/02/2016
00004995 C3             <1>     retn
                        <1>
                        <1> dma_bnd_err_stc:
00004996 F9             <1>     stc
                        <1> dma_bnd_err:
00004997 C605[08770100]09 <1>     mov     byte [DSKETTE_STATUS], DMA_BOUNDARY ; SET ERROR
0000499E C3             <1>     retn                            ; CY SET BY ABOVE IF ERROR
                        <1>
                        <1> ;; 16/12/2014
                        <1> ;;  CLI                             ; DISABLE INTERRUPTS DURING DMA SET-UP
                        <1> ;;  OUT     DMA+12,AL               ; SET THE FIRST/LA5T F/F
                        <1> ;;  ;jmp    $+2                     ; WAIT FOR I/O
                        <1> ;;  IODELAY
                        <1> ;;  OUT     DMA+11,AL               ; OUTPUT THE MODE BYTE
                        <1> ;;  ;SIODELAY
                        <1> ;;      ;cmp       AL,42H                  ; DMA VERIFY COMMAND
                        <1> ;;      ;JNE       short NOT_VERF          ; NO
                        <1> ;;      ;xor       AX,AX                   ; START ADDRESS
                        <1> ;;      ;jmp       SHORT J33
                        <1> ;;;NOT_VERF:
                        <1> ;;  ;mov    AX,ES                   ; GET THE ES VALUE
                        <1> ;;  ;ROL    AX,4                    ; ROTATE LEFT
                        <1> ;;  ;mov    CH,AL                   ; GET HIGHEST NIBBLE OF ES TO CH
                        <1> ;;  ;and    AL,11110000B            ; ZERO THE LOW NIBBLE FROM SEGMENT
                        <1> ;;  ;add    AX,[BP+2]               ; TEST FOR CARRY FROM ADDITION
                        <1> ;;  mov     eax,[ebp+4] ; 06/02/2015
                        <1> ;;  ;jnc    short J33
                        <1> ;;  ;inc    CH                      ; CARRY MEANS HIGH 4 BITS MUST BE INC
                        <1> ;;;J33:
                        <1> ;;  push    eax                     ; SAVE START ADDRESS
                        <1> ;;  OUT     DMA+4,AL                ; OUTPUT LOW ADDRESS
                        <1> ;;  ;jmp    $+2                     ; WAIT FOR I/O
                        <1> ;;  IODELAY
                        <1> ;;  mov     AL,AH
                        <1> ;;  OUT     DMA+4,AL                ; OUTPUT HIGH ADDRESS
                        <1> ;;  shr     eax, 16      ; 07/02/2015
                        <1> ;;  ;mov    AL,CH                   ; GET HIGH 4 BITS
                        <1> ;;  ;jmp    $+2                     ; I/O WAIT STATE
                        <1> ;;  IODELAY
                        <1> ;;  ;and    AL,00001111B
                        <1> ;;  OUT     081H,AL                 ; OUTPUT HIGH 4 BITS TO PAGE REGISTER
                        <1> ;;  ;SIODELAY
                        <1> ;;
                        <1> ;;;----- DETERMINE COUNT
                        <1> ;;  sub     eax, eax ; 08/02/2015
                        <1> ;;  mov     AX,SI                   ; AL =  # OF SECTORS
                        <1> ;;  xchg    AL,AH                   ; AH =  # OF SECTORS
                        <1> ;;  sub     AL,AL                   ; AL = 0, AX = # SECTORS * 256
                        <1> ;;  SHR     AX,1                    ; AX = # SECTORS * 128
                        <1> ;;  push    AX                      ; SAVE # OF SECTORS * 128
                        <1> ;;  mov     DL,3                    ; GET BYTES/SECTOR PARAMETER
                        <1> ;;  call    GET_PARM                ; "
                        <1> ;;  mov     CL,AH                   ; SHIFT COUNT (0=128, 1=256, 2=512 ETC)
                        <1> ;;  pop     AX                      ; AX = # SECTORS * 128
                        <1> ;;  SHL     AX,CL                   ; SHIFT BY PARAMETER VALUE
                        <1> ;;  dec     AX                      ; -1 FOR DMA VALUE
                        <1> ;;  push    eax  ; 08/02/2015       ; SAVE COUNT VALUE
                        <1> ;;  OUT     DMA+5,AL                ; LOW BYTE OF COUNT
                        <1> ;;  ;jmp    $+2                     ; WAIT FOR I/O
                        <1> ;;  IODELAY
                        <1> ;;  mov     AL,AH
                        <1> ;;  OUT     DMA+5,AL                ; HIGH BYTE OF COUNT
                        <1> ;;  ;IODELAY
                        <1> ;;  STI                             ; RE-ENABLE INTERRUPTS
                        <1> ;;  pop     ecx  ; 08/02/2015       ; RECOVER COUNT VALUE
                        <1> ;;  pop     eax  ; 08/02/2015       ; RECOVER ADDRESS VALUE
                        <1> ;;  ;add    AX, CX                  ; ADD, TEST FOR 64K OVERFLOW
                        <1> ;;  add     ecx,eax ; 08/02/2015
                        <1> ;;  mov     AL, 2                   ; MODE FOR 8237
                        <1> ;;  ;jmp    $+2                     ; WAIT FOR I/O
                        <1> ;;  SIODELAY
                        <1> ;;  OUT     DMA+10,AL               ; INITIALIZE THE DISKETTE CHANNEL
                        <1> ;;  ;jnc    short NO_BAD            ; CHECK FOR ERROR
                        <1> ;;  jc      short dma_bnd_err ; 08/02/2015
                        <1> ;;  and     ecx,0FFF00000h ; 16 MB limit
                        <1> ;;  jz      short NO_BAD
                        <1> ;;dma_bnd_err:
                        <1> ;;  mov     byte [DSKETTE_STATUS],DMA_BOUNDARY ; SET ERROR
                        <1> ;;NO_BAD:
                        <1> ;;  retn                            ; CY SET BY ABOVE IF ERROR
                        <1>
                        <1> ;--------------------------------------------------------------------------------
                        <1> ; FMTDMA_SET
                        <1> ;   THIS ROUTINE SETS UP THE DMA CONTROLLER FOR A FORMAT OPERATION.
                        <1> ;
                        <1> ; ON ENTRY: NOTHING REQUIRED
                        <1> ;
                        <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
                        <1> ;--------------------------------------------------------------------------------
                        <1>
                        <1> FMTDMA_SET:
                        <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
                        <1>     ;; 20/02/2015 modification
                        <1>     ;;mov   edx, [ebp+4]            ; Buffer address
                        <1>     ; 06/08/2022
                        <1>     ;mov    edx, ebp ; buffer address
                        <1>     ; 06/08/2022
                        <1>     ;test   edx, 0FF000000h                 ; 16 MB limit
0000499F F7C5000000FF   <1>     test    ebp, 0FF000000h
000049A5 75EF           <1>     jnz     short dma_bnd_err_stc
                        <1>     ;
                        <1>     ;;push  dx                      ; *
                        <1>     ;; 11/04/2021
                        <1>     ; 06/08/2022
                        <1>     ;push   edx
                        <1>     ;mov    dl, 4                   ; SECTORS/TRACK VALUE IN PARM TABLE
                        <1>     ; 06/08/2022
000049A7 B004           <1>     mov     al, 4                   ; SECTORS/TRACK VALUE IN PARM TABLE
000049A9 E85E020000     <1>     call    GET_PARM                ; "
000049AE 88E0           <1>     mov     al, ah                  ; AL = SECTORS/TRACK VALUE
000049B0 28E4           <1>     sub     ah, ah                  ; AX = SECTORS/TRACK VALUE
                        <1>     ;shl    ax, 2                   ; AX = SEC/TRK * 4 (OFFSET C,H,R,N)
                        <1>     ; 06/08/2022
000049B2 C1E002         <1>     shl     eax, 2
                        <1>     ;dec    ax                      ; -1 FOR DMA VALUE
000049B5 48             <1>     dec     eax
                        <1>     ;mov    cx, ax
000049B6 89C1           <1>     mov     ecx, eax
                        <1>     ;;pop   dx                      ; *
                        <1>     ;; 11/04/2021
```

```
                        <1>     ; 06/08/2022
                        <1>     ;pop    edx
                        <1>     ; 06/08/2022
000049B8 B04A           <1>     mov     al, 04Ah
                        <1>     ;
000049BA EB99           <1>     jmp     short NOT_VERF ; 06/08/2022
                        <1>
                        <1> ;; 06/08/2022
                        <1> ;   add     dx, cx                  ; check for overflow
                        <1> ;   jc      short dma_bnd_err
                        <1> ;   ;
                        <1> ;   sub     dx, cx                  ; Restore start address
                        <1> ;   ;
                        <1> ;   ;mov    al, 04Ah                ; WILL WRITE TO THE DISKETTE
                        <1> ;   cli                             ; DISABLE INTERRUPTS DURING DMA SET-UP
                        <1> ;   out     DMA+12, al              ; SET THE FIRST/LA5T F/F
                        <1> ;   IODELAY                         ; WAIT FOR I/O
                        <1> ;   out     DMA+11, al              ; OUTPUT THE MODE BYTE
                        <1> ;   mov     eax, edx                ; Buffer address
                        <1> ;   out     DMA+4, al               ; OUTPUT LOW ADDRESS
                        <1> ;   IODELAY                         ; WAIT FOR I/O
                        <1> ;   mov     al, ah
                        <1> ;   out     DMA+4, al               ; OUTPUT HIGH ADDRESS
                        <1> ;   shr     eax, 16
                        <1> ;   IODELAY                         ; I/O WAIT STATE
                        <1> ;   out     081h, al                ; OUTPUT HIGHEST BITS TO PAGE REGISTER
                        <1> ;   IODELAY
                        <1> ;   ;mov    ax, cx                  ; Byte count - 1
                        <1> ;   ; 06/08/2022
                        <1> ;   mov     eax, ecx
                        <1> ;   out     DMA+5, al               ; LOW BYTE OF COUNT
                        <1> ;   IODELAY                         ; WAIT FOR I/O
                        <1> ;   mov     al, ah
                        <1> ;   out     DMA+5, al               ; HIGH BYTE OF COUNT
                        <1> ;   IODELAY
                        <1> ;   sti                             ; RE-ENABLE INTERRUPTS
                        <1> ;   mov     al, 2                   ; MODE FOR 8237
                        <1> ;   out     DMA+10, al              ; INITIALIZE THE DISKETTE CHANNEL
                        <1> ;
                        <1> ;   ; 06/08/2022
                        <1> ;   clc
                        <1> ;   retn
                        <1>
                        <1> ;; 08/02/2015 - Protected Mode Modification
                        <1> ;;  mov     AL,04AH                 ; WILL WRITE TO THE DISKETTE
                        <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> ;;  ;mov    AX,ES                   ; GET THE ES VALUE
                        <1> ;;  ;ROL    AX,4                    ; ROTATE LEFT
                        <1> ;;  ;mov    CH,AL                   ; GET HIGHEST NIBBLE OF ES TO CH
                        <1> ;;  ;and    AL,11110000B            ; ZERO THE LOW NIBBLE FROM SEGMENT
                        <1> ;;  ;add    AX,[BP+2]               ; TEST FOR CARRY FROM ADDITION
                        <1> ;;  ;jnc    short J33A
                        <1> ;;  ;inc    CH                      ; CARRY MEANS HIGH 4 BITS MUST BE INC
                        <1> ;;  mov     eax,[ebp+4] ; 08/02/2015
                        <1> ;;;J33A:
                        <1> ;;  push    eax ; 08/02/2015        ; SAVE START ADDRESS
                        <1> ;;  OUT     DMA+4,AL                ; OUTPUT LOW ADDRESS
                        <1> ;;  ;jmp    $+2                     ; WAIT FOR I/O
                        <1> ;;  IODELAY
                        <1> ;;  mov     AL,AH
                        <1> ;;  OUT     DMA+4,AL                ; OUTPUT HIGH ADDRESS
                        <1> ;;  shr     eax,16 ; 08/02/2015
                        <1> ;;  ;mov    AL,CH                   ; GET HIGH 4 BITS
                        <1> ;;  ;jmp    $+2                     ; I/O WAIT STATE
                        <1> ;;  IODELAY
                        <1> ;;  ;and    AL,00001111B
                        <1> ;;  OUT     081H,AL                 ; OUTPUT HIGH 4 BITS TO PAGE REGISTER
                        <1> ;;
                        <1> ;;;----- DETERMINE COUNT
                        <1> ;;  sub     eax,eax ; 08/02/2015
                        <1> ;;  mov     DL,4                    ; SECTORS/TRACK VALUE IN PARM TABLE
                        <1> ;;  call    GET_PARM                ; "
                        <1> ;;  xchg    AL,AH                   ; AL = SECTORS/TRACK VALUE
                        <1> ;;  sub     AH,AH                   ; AX = SECTORS/TRACK VALUE
                        <1> ;;  SHL     AX,2                    ; AX = SEC/TRK * 4 (OFFSET C,H,R,N)
                        <1> ;;  dec     AX                      ; -1 FOR DMA VALUE
                        <1> ;;  push    eax     ; 08/02/2015   ; SAVE # OF BYTES TO BE TRANSFERED
                        <1> ;;  OUT     DMA+5,AL                ; LOW BYTE OF COUNT
                        <1> ;;  ;jmp    $+2                     ; WAIT FOR I/O
                        <1> ;;  IODELAY
                        <1> ;;  mov     AL,AH
                        <1> ;;  OUT     DMA+5,AL                ; HIGH BYTE OF COUNT
                        <1> ;;  STI                             ; RE-ENABLE INTERRUPTS
                        <1> ;;  pop     ecx     ; 08/02/2015   ; RECOVER COUNT VALUE
                        <1> ;;  pop     eax     ; 08/02/2015   ; RECOVER ADDRESS VALUE
                        <1> ;;  ;add    AX,CX                   ; ADD, TEST FOR 64K OVERFLOW
                        <1> ;;  add     ecx,eax ; 08/02/2015
                        <1> ;;  mov     AL,2                    ; MODE FOR 8237
                        <1> ;;  ;jmp    $+2                     ; WAIT FOR I/O
                        <1> ;;  SIODELAY
                        <1> ;;  OUT     DMA+10,AL               ; INITIALIZE THE DISKETTE CHANNEL
                        <1> ;;  ;jnc    short FMTDMA_OK                 ; CHECK FOR ERROR
                        <1> ;;  jc      short fmtdma_bnd_err ; 08/02/2015
                        <1> ;;  and     ecx,0FFF00000h ; 16 MB limit
                        <1> ;;  jz      short FMTDMA_OK
                        <1> ;;  stc     ; 20/02/2015
                        <1> ;;fmtdma_bnd_err:
                        <1> ;;  mov     byte [DSKETTE_STATUS],DMA_BOUNDARY ; SET ERROR
                        <1> ;;FMTDMA_OK:
                        <1> ;;  retn                            ; CY SET BY ABOVE IF ERROR
                        <1>
                        <1> ;--------------------------------------------------------------------------------
                        <1> ; NEC_INIT
                        <1> ;   THIS ROUTINE SEEKS TO THE REQUESTED TRACK AND INITIALIZES
                        <1> ;   THE NEC FOR THE READ/WRITE/VERIFY/FORMAT OPERATION.
                        <1> ;
                        <1> ; ON ENTRY: AH = NEC COMMAND TO BE PERFORMED
                        <1> ;
                        <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
                        <1> ;--------------------------------------------------------------------------------
                        <1> NEC_INIT:
                        <1>     ; 11/08/2022
                        <1>     ; EBX = user's ECX register content (on stack)
                        <1>     ; 10/08/2022
                        <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
                        <1>     ;push   ax                      ; SAVE NEC COMMAND
                        <1>     ; 11/04/2021
000049BC 50             <1>     push    eax
000049BD E885020000     <1>     call    MOTOR_ON                ; TURN MOTOR ON FOR SPECIFIC DRIVE
                        <1>
```

```
2458                            <1> ;-----      DO THE SEEK OPERATION
2459                            <1>
2460                            <1>     ;mov    ch, [ebp+1]             ; CH = TRACK #
2461                            <1>     ; 10/08/2022
2462 000049C2 88FD              <1>     mov     ch, bh ; CH = TRACK #
2463 000049C4 E870030000        <1>     call    SEEK                    ; MOVE TO CORRECT TRACK
2464                            <1>     ;pop    ax                      ; RECOVER COMMAND
2465                            <1>     ; 11/04/2021
2466 000049C9 58                <1>     pop     eax
2467 000049CA 721D              <1>     jc      short ER_1              ; ERROR ON SEEK
2468 000049CC BB[E9490000]      <1>     mov     ebx, ER_1               ; LOAD ERROR ADDRESS
2469 000049D1 53                <1>     push    ebx                     ; PUSH NEC_OUT ERROR RETURN
2470                            <1>
2471                            <1> ;-----      SEND OUT THE PARAMETERS TO THE CONTROLLER
2472                            <1>
2473 000049D2 E82B030000        <1>     call    NEC_OUTPUT              ; OUTPUT THE OPERATION COMMAND
2474                            <1>     ;mov    ax, si                  ; AH = HEAD #
2475                            <1>     ; 06/08/2022
2476 000049D7 89F0              <1>     mov     eax, esi
2477 000049D9 89FB              <1>     mov     ebx, edi                ; BL = DRIVE #
2478 000049DB C0E402            <1>     sal     ah, 2                   ; MOVE IT TO BIT 2
2479 000049DE 80E404            <1>     and     ah, 00000100b           ; ISOLATE THAT BIT
2480 000049E1 08DC              <1>     or      ah, bl                  ; OR IN THE DRIVE NUMBER
2481 000049E3 E81A030000        <1>     call    NEC_OUTPUT              ; FALL THRU CY SET IF ERROR
2482 000049E8 5B                <1>     pop     ebx                     ; THROW AWAY ERROR RETURN
2483                            <1> ER_1:
2484 000049E9 C3                <1>     retn
2485                            <1>
2486                            <1> ;--------------------------------------------------------------------------------
2487                            <1> ; RWV_COM
2488                            <1> ;   THIS ROUTINE SENDS PARAMETERS TO THE NEC SPECIFIC TO THE
2489                            <1> ;   READ/WRITE/VERIFY OPERATIONS.
2490                            <1> ;
2491                            <1> ; ON ENTRY: CS:BX = ADDRESS OF MEDIA/DRIVE PARAMETER TABLE
2492                            <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
2493                            <1> ;--------------------------------------------------------------------------------
2494                            <1> RWV_COM:
2495                            <1>     ; 11/08/2022
2496                            <1>     ; 10/08/2022
2497                            <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
2498 000049EA B8[364A0000]      <1>     mov     eax, ER_2               ; LOAD ERROR ADDRESS
2499 000049EF 50                <1>     push    eax                     ; PUSH NEC_OUT ERROR RETURN
2500                            <1>     ;mov    ah, [ebp+1]             ; OUTPUT TRACK #
2501                            <1>     ; 11/08/2022
2502 000049F0 8A642425          <1>     mov     ah, [esp+37] ; CH = OUTPUT TRACK #
2503 000049F4 E809030000        <1>     call    NEC_OUTPUT
2504                            <1>     ;mov    ax, si                  ; OUTPUT HEAD #
2505                            <1>     ; 06/08/2022
2506 000049F9 89F0              <1>     mov     eax, esi
2507 000049FB E802030000        <1>     call    NEC_OUTPUT
2508                            <1>     ;mov    ah, [ebp]               ; OUTPUT SECTOR #
2509                            <1>     ; 11/08/2022
2510 00004A00 8A642424          <1>     mov     ah, [esp+36] ; CL = OUTPUT SECTOR #
2511 00004A04 E8F9020000        <1>     call    NEC_OUTPUT
2512                            <1>     ;mov    dl, 3                   ; BYTES/SECTOR PARAMETER FROM BLOCK
2513                            <1>     ; 06/08/2022
2514 00004A09 B003              <1>     mov     al, 3
2515 00004A0B E8FC010000        <1>     call    GET_PARM                ; ... TO THE NEC
2516 00004A10 E8ED020000        <1>     call    NEC_OUTPUT              ; OUTPUT TO CONTROLLER
2517                            <1>     ;mov    dl, 4                   ; EOT PARAMETER FROM BLOCK
2518                            <1>     ; 06/08/2022
2519 00004A15 B004              <1>     mov     al, 4
2520 00004A17 E8F0010000        <1>     call    GET_PARM                ; ... TO THE NEC
2521 00004A1C E8E1020000        <1>     call    NEC_OUTPUT              ; OUTPUT TO CONTROLLER
2522 00004A21 8A6305            <1>     mov     ah, [ebx+MD.GAP]        ; GET GAP LENGTH
2523                            <1> _R15:
2524 00004A24 E8D9020000        <1>     call    NEC_OUTPUT
2525                            <1>     ;mov    dl, 6                   ; DTL PARAMETER PROM BLOCK
2526                            <1>     ; 06/08/2022
2527 00004A29 B006              <1>     mov     al, 6
2528 00004A2B E8DC010000        <1>     call    GET_PARM                ; ... TO THE NEC
2529 00004A30 E8CD020000        <1>     call    NEC_OUTPUT              ; OUTPUT TO CONTROLLER
2530 00004A35 58                <1>     pop     eax                     ; THROW AWAY ERROR EXIT
2531                            <1> ER_2:
2532 00004A36 C3                <1>     retn
2533                            <1>
2534                            <1> ;--------------------------------------------------------------------------------
2535                            <1> ; NEC_TERM
2536                            <1> ;   THIS ROUTINE WAITS FOR THE OPERATION THEN ACCEPTS THE STATUS
2537                            <1> ;   FROM THE NEC FOR THE READ/WRITE/VERIFY/FORWAT OPERATION.
2538                            <1> ;
2539                            <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
2540                            <1> ;--------------------------------------------------------------------------------
2541                            <1> NEC_TERM:
2542                            <1>
2543                            <1> ;-----      LET THE OPERATION HAPPEN
2544                            <1>
2545 00004A37 56                <1>     push    esi                     ; SAVE HEAD #, # OF SECTORS
2546 00004A38 E8CD030000        <1>     call    WAIT_INT                ; WAIT FOR THE INTERRUPT
2547 00004A3D 9C                <1>     pushf
2548 00004A3E E8F6030000        <1>     call    RESULTS                 ; GET THE NEC STATUS
2549 00004A43 724B              <1>     jc      short SET_END_POP
2550 00004A45 9D                <1>     popf
2551 00004A46 723E              <1>     jc      short SET_END           ; LOOK FOR ERROR
2552                            <1>
2553                            <1> ;-----      CHECK THE RESULTS RETURNED BY THE CONTROLLER
2554                            <1>
2555 00004A48 FC                <1>     cld                             ; SET THE CORRECT DIRECTION
2556 00004A49 BE[09770100]      <1>     mov     esi, NEC_STATUS                 ; POINT TO STATUS FIELD
2557 00004A4E AC                <1>     lodsb                           ; GET ST0
2558 00004A4F 24C0              <1>     and     AL, 11000000B           ; TEST FOR NORMAL TERMINATION
2559 00004A51 7433              <1>     jz      short SET_END
2560 00004A53 3C40              <1>     cmp     AL,01000000B            ; TEST FOR ABNORMAL TERMINATION
2561 00004A55 7527              <1>     jnz     short J18               ; NOT ABNORMAL, BAD NEC
2562                            <1>
2563                            <1> ;-----      ABNORMAL TERMINATION, FIND OUT WHY
2564                            <1>
2565 00004A57 AC                <1>     lodsb                           ; GET ST1
2566 00004A58 D0E0              <1>     sal     al, 1                   ; TEST FOR EDT FOUND
2567 00004A5A B404              <1>     mov     ah, RECORD_NOT_FND
2568 00004A5C 7222              <1>     jc      short J19
2569 00004A5E C0E002            <1>     sal     al, 2
2570 00004A61 B410              <1>     mov     ah, BAD_CRC
2571 00004A63 721B              <1>     jc      short J19
2572 00004A65 D0E0              <1>     sal     al, 1                   ; TEST FOR DMA OVERRUN
2573 00004A67 B408              <1>     mov     ah, BAD_DMA
2574 00004A69 7215              <1>     jc      short J19
2575 00004A6B C0E002            <1>     sal     al, 2                   ; TEST FOR RECORD NOT FOUND
2576 00004A6E B404              <1>     mov     ah, RECORD_NOT_FND
2577 00004A70 720E              <1>     jc      short J19
2578 00004A72 D0E0              <1>     sal     al, 1
2579 00004A74 B403              <1>     mov     ah, WRITE_PROTECT       ; TEST FOR WRITE_PROTECT
2580 00004A76 7208              <1>     jc      short J19
2581 00004A78 D0E0              <1>     sal     al, 1                   ; TEST MISSING ADDRESS MARK
```

```
2582 00004A7A B402                <1>     mov     ah, BAD_ADDR_MARK
2583 00004A7C 7202                <1>     jc      short J19
2584                              <1>
2585                              <1> ;-----      NEC MUST HAVE FAILED
2586                              <1> J18:
2587 00004A7E B420                <1>     mov     ah, BAD_NEC
2588                              <1> J19:
2589 00004A80 0825[08770100]      <1>     or      [DSKETTE_STATUS], ah
2590                              <1> SET_END:
2591 00004A86 803D[08770100]01    <1>     cmp     byte [DSKETTE_STATUS], 1 ; SET ERROR CONDITION
2592 00004A8D F5                  <1>     cmc
2593 00004A8E 5E                  <1>     pop     esi
2594 00004A8F C3                  <1>     retn                            ; RESTORE HEAD #, # OF SECTORS
2595                              <1>
2596                              <1> SET_END_POP:
2597 00004A90 9D                  <1>     popf
2598 00004A91 EBF3                <1>     jmp     short SET_END
2599                              <1>
2600                              <1> ;--------------------------------------------------------------------------------
2601                              <1> ; DSTATE:   ESTABLISH STATE UPON SUCCESSFUL OPERATION.
2602                              <1> ;--------------------------------------------------------------------------------
2603                              <1> DSTATE:
2604 00004A93 803D[08770100]00    <1>     cmp     byte [DSKETTE_STATUS], 0 ; CHECK FOR ERROR
2605 00004A9A 753E                <1>     jnz     short SETBAC            ; IF ERROR JUMP
2606 00004A9C 808F[13770100]10    <1>     or      byte [DSK_STATE+edi], MED_DET
2607                              <1>                                     ; NO ERROR, MARK MEDIA AS DETERMINED
2608 00004AA3 F687[13770100]04    <1>     test    byte [DSK_STATE+edi], DRV_DET ; DRIVE DETERMINED ?
2609 00004AAA 752E                <1>     jnz     short SETBAC            ; IF DETERMINED NO TRY TO DETERMINE
2610 00004AAC 8A87[13770100]      <1>     mov     al, [DSK_STATE+edi]     ; LOAD STATE
2611 00004AB2 24C0                <1>     and     al, RATE_MSK            ; KEEP ONLY RATE
2612 00004AB4 3C80                <1>     cmp     al, RATE_250            ; RATE 250 ?
2613 00004AB6 751B                <1>     jne     short M_12              ; NO, MUST BE 1.2M OR 1.44M DRIVE
2614                              <1>
2615                              <1> ;-----      CHECK IF IT IS 1.44M
2616                              <1>
2617 00004AB8 E846010000          <1>     call    CMOS_TYPE               ; RETURN DRIVE TYPE IN (AL)
2618                              <1>     ;;20/02/2015
2619                              <1>     ;;jc    short M_12              ; CMOS BAD
2620 00004ABD 7414                <1>     jz      short M_12 ;; 20/02/2015
2621 00004ABF 3C04                <1>     cmp     al, 4                   ; 1.44MB DRIVE ?
2622 00004AC1 7410                <1>     je      short M_12              ; YES
2623                              <1> M_720:
2624 00004AC3 80A7[13770100]FD    <1>     and     byte [DSK_STATE+edi], ~FMT_CAPA ; TURN OFF FORMAT CAPABILITY
2625 00004ACA 808F[13770100]04    <1>     or      byte [DSK_STATE+edi], DRV_DET ; MARK DRIVE DETERMINED
2626 00004AD1 EB07                <1>     jmp     short SETBAC            ; BACK
2627                              <1> M_12:
2628 00004AD3 808F[13770100]06    <1>     or      byte [DSK_STATE+edi], DRV_DET+FMT_CAPA
2629                              <1>                                     ; TURN ON DETERMINED & FMT CAPA
2630                              <1> SETBAC:
2631 00004ADA C3                  <1>     retn
2632                              <1>
2633                              <1> ;--------------------------------------------------------------------------------
2634                              <1> ; RETRY
2635                              <1> ;   DETERMINES WHETHER A RETRY IS NECESSARY.
2636                              <1> ;   IF RETRY IS REQUIRED THEN STATE INFORMATION IS UPDATED FOR RETRY.
2637                              <1> ;
2638                              <1> ; ON EXIT:  CY = 1 FOR RETRY, CY = 0 FOR NO RETRY
2639                              <1> ;--------------------------------------------------------------------------------
2640                              <1> RETRY:
2641                              <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
2642 00004ADB 803D[08770100]00    <1>     cmp     byte [DSKETTE_STATUS], 0 ; GET STATUS OF OPERATION
2643 00004AE2 7445                <1>     jz      short NO_RETRY          ; SUCCESSFUL OPERATION
2644 00004AE4 803D[08770100]80    <1>     cmp     byte [DSKETTE_STATUS], TIME_OUT ; IF TIME OUT NO RETRY
2645 00004AEB 743C                <1>     jz      short NO_RETRY
2646 00004AED 8AA7[13770100]      <1>     mov     ah, [DSK_STATE+edi]     ; GET MEDIA STATE OF DRIVE
2647 00004AF3 F6C410              <1>     test    ah, MED_DET             ; ESTABLISHED/DETERMINED ?
2648 00004AF6 7531                <1>     jnz     short NO_RETRY          ; IF ESTABLISHED STATE THEN TRUE ERROR
2649 00004AF8 80E4C0              <1>     and     ah, RATE_MSK            ; ISOLATE RATE
2650 00004AFB 8A2D[10770100]      <1>     mov     ch, [LASTRATE]          ; GET START OPERATION STATE
2651 00004B01 C0C504              <1>     rol     ch, 4                   ; TO CORRESPONDING BITS
2652 00004B04 80E5C0              <1>     and     ch, RATE_MSK            ; ISOLATE RATE BITS
2653 00004B07 38E5                <1>     cmp     ch, ah                  ; ALL RATES TRIED
2654 00004B09 741E                <1>     je      short NO_RETRY          ; IF YES, THEN TRUE ERROR
2655                              <1>
2656                              <1> ;   SETUP STATE INDICATOR FOR RETRY ATTEMPT TO NEXT RATE
2657                              <1> ;    00000000B (500) -> 10000000B (250)
2658                              <1> ;    10000000B (250) -> 01000000B (300)
2659                              <1> ;    01000000B (300) -> 00000000B (500)
2660                              <1>
2661 00004B0B 80FC01              <1>     cmp     ah, RATE_500+1          ; SET CY FOR RATE 500
2662 00004B0E D0DC                <1>     rcr     ah, 1                   ; TO NEXT STATE
2663 00004B10 80E4C0              <1>     and     ah, RATE_MSK            ; KEEP ONLY RATE BITS
2664 00004B13 80A7[13770100]1F    <1>     and     byte [DSK_STATE+edi], ~(RATE_MSK+DBL_STEP)
2665                              <1>                                     ; RATE, DBL STEP OFF
2666 00004B1A 08A7[13770100]      <1>     or      [DSK_STATE+edi], ah     ; TURN ON NEW RATE
2667 00004B20 C605[08770100]00    <1>     mov     byte [DSKETTE_STATUS], 0  ; RESET STATUS FOR RETRY
2668 00004B27 F9                  <1>     stc                             ; SET CARRY FOR RETRY
2669 00004B28 C3                  <1>     retn                            ; RETRY RETURN
2670                              <1>
2671                              <1> NO_RETRY:
2672                              <1>     ; 06/08/2022
2673                              <1>     ;clc                            ; CLEAR CARRY NO RETRY
2674 00004B29 C3                  <1>     retn                            ; NO RETRY RETURN
2675                              <1>
2676                              <1> ;--------------------------------------------------------------------------------
2677                              <1> ; NUM_TRANS
2678                              <1> ;   THIS ROUTINE CALCULATES THE NUMBER OF SECTORS THAT WERE
2679                              <1> ;   ACTUALLY TRANSFERRED TO/FROM THE DISKETTE.
2680                              <1> ;
2681                              <1> ; ON ENTRY: [BP+1] = TRACK
2682                              <1> ;           SI-HI  = HEAD
2683                              <1> ;           [BP]   = START SECTOR
2684                              <1> ;
2685                              <1> ; ON EXIT:  AL = NUMBER ACTUALLY TRANSFERRED
2686                              <1> ;--------------------------------------------------------------------------------
2687                              <1> NUM_TRANS:
2688                              <1>     ; 10/08/2022
2689                              <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
2690 00004B2A 30C0                <1>     xor     al, al                  ; CLEAR FOR ERROR
2691                              <1>     ;cmp    byte [DSKETTE_STATUS], 0
2692 00004B2C 3805[08770100]      <1>     cmp     [DSKETTE_STATUS], al ; 0 ; CHECK FOR ERROR
2693 00004B32 752D                <1>     jnz     short NT_OUT            ; IF ERROR 0 TRANSFERRED
2694                              <1>     ;mov    dl, 4                   ; SECTORS/TRACK OFFSET TO DL
2695                              <1>     ; 06/08/2022
2696 00004B34 B004                <1>     mov     al, 4
2697 00004B36 E8D1000000          <1>     call    GET_PARM                ; AH = SECTORS/TRACK
2698 00004B3B 8A1D[0E770100]      <1>     mov     bl, [NEC_STATUS+5]      ; GET ENDING SECTOR
2699                              <1>     ;mov    cx, si                  ; CH = HEAD # STARTED
2700                              <1>     ; 06/08/2022
2701 00004B41 89F1                <1>     mov     ecx, esi
2702 00004B43 3A2D[0D770100]      <1>     cmp     ch, [NEC_STATUS+4]      ; GET HEAD ENDED UP ON
2703 00004B49 750E                <1>     jnz     short DIF_HD            ; IF ON SAME HEAD, THEN NO ADJUST
2704 00004B4B 8A2D[0C770100]      <1>     mov     ch, [NEC_STATUS+3]      ; GET TRACK ENDED UP ON
2705                              <1>     ;cmp    ch, [ebp+1]             ; IS IT ASKED FOR TRACK
```

```
                                <1>     ; 10/08/2022
00004B51 3A6C241D               <1>     cmp     ch, [esp+29] ; CH = TRACK #
00004B55 7404                   <1>     jz      short SAME_TRK          ; IF SAME TRACK NO INCREASE
00004B57 00E3                   <1>     add     bl, ah                  ; ADD SECTORS/TRACK
                                <1> DIF_HD:
00004B59 00E3                   <1>     add     bl, ah                  ; ADD SECTORS/TRACK
                                <1> SAME_TRK:
                                <1>     ;sub    bl, [ebp]               ; SUBTRACT START FROM END
                                <1>     ; 10/08/2022
00004B5B 2A5C241C               <1>     sub     bl, [esp+28] ; CL = SECTOR #
00004B5F 88D8                   <1>     mov     al, bl                  ; TO AL
                                <1> NT_OUT:
00004B61 C3                     <1>     retn
                                <1>
                                <1> ;--------------------------------------------------------------------------------
                                <1> ; SETUP_DBL
                                <1> ;   CHECK DOUBLE STEP.
                                <1> ;
                                <1> ; ON ENTRY :       DI = DRIVE
                                <1> ;
                                <1> ; ON EXIT : CY = 1 MEANS ERROR
                                <1> ;--------------------------------------------------------------------------------
                                <1> SETUP_DBL:
                                <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
00004B62 8AA7[13770100]         <1>     mov     ah, [DSK_STATE+edi]     ; ACCESS STATE
00004B68 F6C410                 <1>     test    ah, MED_DET             ; ESTABLISHED STATE ?
00004B6B 7578                   <1>     jnz     short NO_DBL            ; IF ESTABLISHED THEN DOUBLE DONE
                                <1>
                                <1> ;-----      CHECK FOR TRACK 0 TO SPEED UP ACKNOWLEDGE OF UNFORMATTED DISKETTE
                                <1>
00004B6D C605[05770100]00       <1>     mov     byte [SEEK_STATUS], 0 ; SET RECALIBRATE REQUIRED ON ALL DRIVES
00004B74 E8CE000000             <1>     call    MOTOR_ON                ; ENSURE MOTOR STAY ON
00004B79 B500                   <1>     mov     ch, 0                   ; LOAD TRACK 0
00004B7B E8B9010000             <1>     call    SEEK                    ; SEEK TO TRACK 0
00004B80 E861000000             <1>     call    READ_ID                 ; READ ID FUNCTION
00004B85 7243                   <1>     jc      short SD_ERR            ; IF ERROR NO TRACK 0
                                <1>
                                <1> ;-----      INITIALIZE START AND MAX TRACKS (TIMES 2 FOR BOTH HEADS)
                                <1>
00004B87 66B95004               <1>     mov     cx, 0450h               ; START, MAX TRACKS
00004B8B F687[13770100]01       <1>     test    byte [DSK_STATE+edi], TRK_CAPA ; TEST FOR 80 TRACK CAPABILITY
00004B92 7402                   <1>     jz      short CNT_OK            ; IF NOT COUNT IS SETUP
00004B94 B1A0                   <1>     mov     cl, 0A0h                ; MAXIMUM TRACK 1.2 MB
                                <1>
                                <1> ;   ATTEMPT READ ID OF ALL TRACKS, ALL HEADS UNTIL SUCCESS; UPON SUCCESS,
                                <1> ;   MUST SEE IF ASKED FOR TRACK IN SINGLE STEP MODE = TRACK ID READ; IF NOT
                                <1> ;   THEN SET DOUBLE STEP ON.
                                <1>
                                <1> CNT_OK:
                                <1>     ; 11/04/2021 (32 bit push/pop)
00004B96 C605[07770100]FF       <1>         mov     byte [MOTOR_COUNT], 0FFh ; ENSURE MOTOR STAYS ON FOR OPERATION
00004B9D 51                     <1>     push    ecx                     ; SAVE TRACK, COUNT
00004B9E C605[08770100]00       <1>     mov     byte [DSKETTE_STATUS], 0 ; CLEAR STATUS, EXPECT ERRORS
                                <1>     ;xor    ax, ax                  ; CLEAR AX
                                <1>     ; 06/08/2022
00004BA5 31C0                   <1>     xor     eax, eax
00004BA7 D0ED                   <1>     shr     ch, 1                   ; HALVE TRACK, CY = HEAD
00004BA9 C0D003                 <1>     rcl     al, 3                   ; AX = HEAD IN CORRECT BIT
00004BAC 50                     <1>     push    eax                     ; SAVE HEAD
00004BAD E887010000             <1>     call    SEEK                    ; SEEK TO TRACK
00004BB2 58                     <1>     pop     eax                     ; RESTORE HEAD
                                <1>     ;or     di, ax                  ; DI = HEAD OR'ED DRIVE
                                <1>     ; 06/08/2022
00004BB3 09C7                   <1>     or      edi, eax
00004BB5 E82C000000             <1>     call    READ_ID                 ; READ ID HEAD 0
00004BBA 9C                     <1>     pushf                           ; SAVE RETURN FROM READ_ID
00004BBB 6681E7FB00             <1>     and     di, 11111011b           ; TURN OFF HEAD 1 BIT
00004BC0 9D                     <1>     popf                            ; RESTORE ERROR RETURN
00004BC1 59                     <1>     pop     ecx                     ; RESTORE COUNT
00004BC2 7308                   <1>     jnc     short DO_CHK            ; IF OK, ASKED = RETURNED TRACK ?
00004BC4 FEC5                   <1>     inc     ch                      ; INC FOR NEXT TRACK
00004BC6 38CD                   <1>     cmp     ch, cl                  ; REACHED MAXIMUM YET
00004BC8 75CC                   <1>     jnz     short CNT_OK            ; CONTINUE TILL ALL TRIED
                                <1>
                                <1> ;-----      FALL THRU, READ ID FAILED FOR ALL TRACKS
                                <1>
                                <1> SD_ERR:
00004BCA F9                     <1>     stc                             ; SET CARRY FOR ERROR
00004BCB C3                     <1>     retn                            ; SETUP_DBL ERROR EXIT
                                <1>
                                <1> DO_CHK:
00004BCC 8A0D[0C770100]         <1>     mov     cl, [NEC_STATUS+3]      ; LOAD RETURNED TRACK
00004BD2 888F[15770100]         <1>     mov     [DSK_TRK+edi], cl       ; STORE TRACK NUMBER
00004BD8 D0ED                   <1>     shr     ch, 1                   ; HALVE TRACK
00004BDA 38CD                   <1>     cmp     ch, cl                  ; IS IT THE SAME AS ASKED FOR TRACK
00004BDC 7407                   <1>     jz      short NO_DBL            ; IF SAME THEN NO DOUBLE STEP
00004BDE 808F[13770100]20       <1>     or      byte [DSK_STATE+edi], DBL_STEP ; TURN ON DOUBLE STEP REQUIRED
                                <1> NO_DBL:
                                <1>     ; 06/08/2022
                                <1>     ;clc                            ; CLEAR ERROR FLAG
00004BE5 C3                     <1>     retn
                                <1>
                                <1> ;--------------------------------------------------------------------------------
                                <1> ; READ_ID
                                <1> ;   READ ID FUNCTION.
                                <1> ;
                                <1> ; ON ENTRY: DI : BIT 2 = HEAD; BITS 1,0 = DRIVE
                                <1> ;
                                <1> ; ON EXIT:  DI : BIT 2 IS RESET, BITS 1,0 = DRIVE
                                <1> ;           @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
                                <1> ;--------------------------------------------------------------------------------
                                <1> READ_ID:
                                <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
00004BE6 B8[024C0000]           <1>     mov     eax, ER_3               ; MOVE NEC OUTPUT ERROR ADDRESS
00004BEB 50                     <1>     push    eax
00004BEC B44A                   <1>     mov     ah, 4Ah                 ; READ ID COMMAND
00004BEE E80F010000             <1>     call    NEC_OUTPUT              ; TO CONTROLLER
                                <1>     ;mov    ax, di                  ; DRIVE # TO AH, HEAD 0
                                <1>     ; 06/08/2022
00004BF3 89F8                   <1>     mov     eax, edi
00004BF5 88C4                   <1>     mov     ah, al
00004BF7 E806010000             <1>     call    NEC_OUTPUT              ; TO CONTROLLER
00004BFC E836FEFFFF             <1>     call    NEC_TERM                ; WAIT FOR OPERATION, GET STATUS
00004C01 58                     <1>     pop     eax                     ; THROW AWAY ERROR ADDRESS
                                <1> ER_3:
00004C02 C3                     <1>     retn
                                <1>
                                <1> ;--------------------------------------------------------------------------------
                                <1> ; CMOS_TYPE
                                <1> ;   RETURNS DISKETTE TYPE FROM CMOS
                                <1> ;
                                <1> ; ON ENTRY: DI = DRIVE #
                                <1> ;
                                <1> ; ON EXIT:  AL = TYPE; CY REFLECTS STATUS
```

```
2830                                    <1> ;---------------------------------------------------------------------------
2831                                    <1>
2832                                    <1> CMOS_TYPE: ; 11/12/2014
2833 00004C03 8A87[7C660000]            <1>     mov     al, [edi+fd0_type]
2834 00004C09 20C0                      <1>     and     al, al ; 18/12/2014
2835 00004C0B C3                        <1>     retn
2836                                    <1>
2837                                    <1> ;CMOS_TYPE:
2838                                    <1> ;   mov     al, CMOS_DIAG           ; CMOS DIAGNOSTIC STATUS BYTE ADDRESS
2839                                    <1> ;   call    CMOS_READ               ; GET CMOS STATUS
2840                                    <1> ;   test    al, BAD_BAT+BAD_CKSUM   ; BATTERY GOOD AND CHECKSUM VALID
2841                                    <1> ;   stc                             ; SET CY = 1 INDICATING ERROR FOR RETURN
2842                                    <1> ;   jnz     short BAD_CM            ; ERROR IF EITHER BIT ON
2843                                    <1> ;   mov     al, CMOS_DISKETTE       ; ADDRESS OF DISKETTE BYTE IN CMOS
2844                                    <1> ;   call    CMOS_READ               ; GET DISKETTE BYTE
2845                                    <1> ;   or      di, di                  ; SEE WHICH DRIVE IN QUESTION
2846                                    <1> ;   jnz     short TB                ; IF DRIVE 1, DATA IN LOW NIBBLE
2847                                    <1> ;   ror     al, 4                   ; EXCHANGE NIBBLES IF SECOND DRIVE
2848                                    <1> ;TB:
2849                                    <1> ;   and     al, 0Fh                 ; KEEP ONLY DRIVE DATA, RESET CY, 0
2850                                    <1> ;BAD_CM:
2851                                    <1> ;   retn                            ; CY, STATUS OF READ
2852                                    <1>
2853                                    <1> ;---------------------------------------------------------------------------
2854                                    <1> ; GET_PARM
2855                                    <1> ;   THIS ROUTINE FETCHES THE INDEXED POINTER FROM THE DISK_BASE
2856                                    <1> ;   BLOCK POINTED TO BY THE DATA VARIABLE @DISK_POINTER. A BYTE FROM
2857                                    <1> ;   THAT TABLE IS THEN MOVED INTO AH, THE INDEX OF THAT BYTE BEING
2858                                    <1> ;   THE PARAMETER IN DL.
2859                                    <1> ;
2860                                    <1> ; ON ENTRY: DL = INDEX OF BYTE TO BE FETCHED
2861                                    <1> ;
2862                                    <1> ; ON EXIT:  AH = THAT BYTE FROM BLOCK
2863                                    <1> ;           AL, DH DESTROYED
2864                                    <1> ;---------------------------------------------------------------------------
2865                                    <1> GET_PARM:
2866                                    <1>     ; 09/08/2022
2867                                    <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
2868                                    <1>     ; AL = INDEX
2869                                    <1>     ; EDI = (current) DRIVE #
2870                                    <1>     ;;push  ds
2871                                    <1>     ;push   esi
2872                                    <1>     ;;sub   ax, ax                  ; DS = 0, BIOS DATA AREA
2873                                    <1>     ;;mov   ds, ax
2874                                    <1>     ;;mov   ax, cs
2875                                    <1>     ;;mov   ds, ax
2876                                    <1>     ; 08/02/2015 (protected mode modifications, bx -> ebx)
2877                                    <1>     ;xchg   edx, ebx                ; BL = INDEX
2878                                    <1>     ; 06/08/2022
2879 00004C0C 53                       <1>     push    ebx                     ; SAVE EBX
2880                                    <1>     ;movzx  ebx, dl                 ; EBX = INDEX
2881 00004C0D 0FB6D8                   <1>     movzx   ebx, al ; 06/08/2022
2882                                    <1>     ;;sub   bh, bh                  ; BX = INDEX
2883                                    <1>     ;and    ebx, 0FFh
2884                                    <1>     ;;lds   si, [DISK_POINTER]      ; POINT TO BLOCK
2885                                    <1>     ;
2886                                    <1>     ; 17/12/2014
2887                                    <1>     ;mov    ax, [cfd]               ; current (AL) and previous fd (AH)
2888                                    <1>     ; 06/08/2022
2889 00004C10 89F8                     <1>     mov     eax, edi                ; EDI = DRIVE #
2890                                    <1>     ;cmp    al, ah
2891 00004C12 3A05[6D660000]           <1>     cmp     al, [pfd]
2892 00004C18 7423                     <1>     je      short gpndc
2893 00004C1A A2[6D660000]             <1>     mov     [pfd], al               ; current drive -> previous drive
2894 00004C1F 53                       <1>     push    ebx ; 08/02/2015
2895                                    <1>     ;mov    bl, al
2896                                    <1>     ;; 11/12/2014
2897                                    <1>     ;mov    al, [ebx+fd0_type]      ; Drive type (0,1,2,3,4)
2898                                    <1>     ; 09/08/2022
2899 00004C20 8A87[7C660000]           <1>     mov     al, [edi+fd0_type]      ; Drive type (0,1,2,3,4)
2900                                    <1>     ; 18/12/2014
2901 00004C26 20C0                     <1>     and     al, al
2902 00004C28 7507                     <1>     jnz     short gpdtc
2903 00004C2A BB[57660000]             <1>     mov     ebx, MD_TBL6            ; 1.44 MB param. tbl. (default)
2904 00004C2F EB05                     <1>         jmp     short gpdpu
2905                                    <1> gpdtc:
2906 00004C31 E809FBFFFF               <1>     call    DR_TYPE_CHECK
2907                                    <1>     ; cf = 1 -> ebx points to 1.44MB fd parameter table (default)
2908                                    <1> gpdpu:
2909 00004C36 891D[F4650000]           <1>     mov     [DISK_POINTER], ebx
2910 00004C3C 5B                       <1>     pop     ebx
2911                                    <1> gpndc:
2912                                    <1>     ;mov    esi, [DISK_POINTER] ; 08/02/2015, si -> esi
2913                                    <1>     ; 06/08/2022
2914 00004C3D 031D[F4650000]           <1>     add     ebx, [DISK_POINTER]
2915                                    <1>     ;mov    ah, [esi+ebx]           ; GET THE WORD
2916 00004C43 8A23                     <1>     mov     ah, [ebx]
2917                                    <1>     ;xchg   edx, ebx                ; RESTORE BX
2918                                    <1>     ; 06/08/2022
2919 00004C45 5B                       <1>     pop     ebx                     ; RESTORE EBX
2920                                    <1>     ;pop    esi
2921                                    <1>     ;;pop   ds
2922 00004C46 C3                       <1>     retn
2923                                    <1>
2924                                    <1> ;---------------------------------------------------------------------------
2925                                    <1> ; MOTOR_ON
2926                                    <1> ;   TURN MOTOR ON AND WAIT FOR MOTOR START UP TIME. THE @MOTOR_COUNT
2927                                    <1> ;   IS REPLACED WITH A SUFFICIENTLY HIGH NUMBER (0FFH) TO ENSURE
2928                                    <1> ;   THAT THE MOTOR DOES NOT GO OFF DURING THE OPERATION. IF THE
2929                                    <1> ;   MOTOR NEEDED TO BE TURNED ON, THE MULTI-TASKING HOOK FUNCTION
2930                                    <1> ;   (AX=90FDH, INT 15) IS CALLED TELLING THE OPERATING SYSTEM
2931                                    <1> ;   THAT THE BIOS IS ABOUT TO WAIT FOR MOTOR START UP. IF THIS
2932                                    <1> ;   FUNCTION RETURNS WITH CY = 1, IT MEANS THAT THE MINIMUM WAIT
2933                                    <1> ;   HAS BEEN COMPLETED. AT THIS POINT A CHECK IS MADE TO ENSURE
2934                                    <1> ;   THAT THE MOTOR WASN'T TURNED OFF BY THE TIMER. IF THE HOOK DID
2935                                    <1> ;   NOT WAIT, THE WAIT FUNCTION (AH=086H) IS CALLED TO WAIT THE
2936                                    <1> ;   PRESCRIBED AMOUNT OF TIME. IF THE CARRY FLAG IS SET ON RETURN,
2937                                    <1> ;   IT MEANS THAT THE FUNCTION IS IN USE AND DID NOT PERFORM THE
2938                                    <1> ;   WAIT. A TIMER 1 WAIT LOOP WILL THEN DO THE WAIT.
2939                                    <1> ;
2940                                    <1> ; ON ENTRY: DI = DRIVE #
2941                                    <1> ; ON EXIT:  AX,CX,DX DESTROYED
2942                                    <1> ;---------------------------------------------------------------------------
2943                                    <1> MOTOR_ON:
2944                                    <1>     ; 06/08/2022 - TRDOS 386 Kernel v2.0.5
2945 00004C47 53                       <1>     push    ebx                     ; SAVE REG.
2946 00004C48 E825000000               <1>     call    TURN_ON                 ; TURN ON MOTOR
2947 00004C4D 7221                     <1>     jc      short MOT_IS_ON                 ; IF CY=1 NO WAIT
2948                                    <1>     ; 06/08/2022
2949                                    <1>     ;call   XLAT_OLD                ; TRANSLATE STATE TO COMPATIBLE MODE
2950 00004C4F E851FBFFFF               <1>     call    XLAT_NEW                ; TRANSLATE STATE TO PRESENT ARCH,
2951                                    <1>     ;call   TURN_ON                 ; CHECK AGAIN IF MOTOR ON
2952                                    <1>     ;jc     short MOT_IS_ON                 ; IF NO WAIT MEANS IT IS ON
2953                                    <1> M_WAIT:
```

```
                        <1>     ;mov    dl, 10                  ; GET THE MOTOR WAIT PARAMETER
                        <1>     ; 06/08/2022
00004C54 B00A           <1>     mov     al, 10
00004C56 E8B1FFFFFF     <1>     call    GET_PARM
                        <1>     ;mov    al, ah                  ; AL = MOTOR WAIT PARAMETER
                        <1>     ;xor    ah, ah                  ; AX = MOTOR WAIT PARAMETER
                        <1>     ;cmp    al, 8                   ; SEE IF AT LEAST A SECOND IS SPECIFIED
00004C5B 80FC08         <1>     cmp     ah, 8
                        <1>     ;jae    short GP2               ; IF YES, CONTINUE
00004C5E 7702           <1>     ja      short J13
                        <1>     ;mov    al, 8                   ; ONE SECOND WAIT FOR MOTOR START UP
00004C60 B408           <1>     mov     ah, 8
                        <1>
                        <1> ;-----      AS CONTAINS NUMBER OF 1/8 SECONDS (125000 MICROSECONDS) TO WAIT
                        <1> GP2:
                        <1> ;-----      FOLLOWING LOOPS REQUIRED WHEN RTC WAIT FUNCTION IS ALREADY IN USE
                        <1> J13:                                ; WAIT FOR 1/8 SECOND PER (AL)
                        <1>     ;mov    ecx, 8286               ; COUNT FOR 1/8 SECOND AT 15.085737 US
                        <1>     ; 11/04/2021
00004C62 B947100000     <1>     mov     ecx, 4167 ; count of 30 micro seconds * (1/8)
00004C67 E8B5D7FFFF     <1>     call    WAITF                   ; GO TO FIXED WAIT ROUTINE
                        <1>     ;dec    al                      ; DECREMENT TIME VALUE
00004C6C FECC           <1>     dec     ah
00004C6E 75F2           <1>     jnz     short J13               ; ARE WE DONE YET
                        <1> MOT_IS_ON:
00004C70 5B             <1>     pop     ebx                     ; RESTORE REG.
00004C71 C3             <1>     retn
                        <1>
                        <1> ;--------------------------------------------------------------------------------
                        <1> ; TURN_ON
                        <1> ;   TURN MOTOR ON AND RETURN WAIT STATE.
                        <1> ;
                        <1> ; ON ENTRY: DI = DRIVE #
                        <1> ;
                        <1> ; ON EXIT:  CY = 0 MEANS WAIT REQUIRED
                        <1> ;           CY = 1 MEANS NO WAIT REQUIRED
                        <1> ;           AX,BX,CX,DX DESTROYED
                        <1> ;--------------------------------------------------------------------------------
                        <1> TURN_ON:
00004C72 89FB           <1>     mov     ebx, edi                ; BX = DRIVE #
00004C74 88D9           <1>     mov     cl, bl                  ; CL = DRIVE #
00004C76 C0C304         <1>     rol     bl, 4                   ; BL = DRIVE SELECT
00004C79 FA             <1>     cli                             ; NO INTERRUPTS WHILE DETERMINING STATUS
00004C7A C605[07770100]FF   <1>     mov     byte [MOTOR_COUNT], 0FFh ; ENSURE MOTOR STAYS ON FOR OPERATION
00004C81 A0[06770100]       <1>     mov     al, [MOTOR_STATUS]      ; GET DIGITAL OUTPUT REGISTER REFLECTION
00004C86 2430           <1>     and     al, 00110000b           ; KEEP ONLY DRIVE SELECT BITS
00004C88 B401           <1>     mov     ah, 1                   ; MASK FOR DETERMINING MOTOR BIT
00004C8A D2E4           <1>     shl     ah, cl                  ; AH = MOTOR ON, A=00000001, B=00000010
                        <1>
                        <1> ;  AL = DRIVE SELECT FROM @MOTOR_STATUS
                        <1> ;  BL = DRIVE SELECT DESIRED
                        <1> ;  AH = MOTOR ON MASK DESIRED
                        <1>
00004C8C 38D8           <1>     cmp     al, bl                  ; REQUESTED DRIVE ALREADY SELECTED ?
00004C8E 7508           <1>     jnz     short TURN_IT_ON        ; IF NOT SELECTED JUMP
00004C90 8425[06770100]     <1>     test    ah, [MOTOR_STATUS]      ; TEST MOTOR ON BIT
00004C96 7535           <1>     jnz     short NO_MOT_WAIT       ; JUMP IF MOTOR ON AND SELECTED
                        <1>
                        <1> TURN_IT_ON:
00004C98 08DC           <1>     or      ah, bl                  ; AH = DRIVE SELECT AND MOTOR ON
00004C9A 8A3D[06770100]     <1>     mov     bh, [MOTOR_STATUS]      ; SAVE COPY OF @MOTOR_STATUS BEFORE
00004CA0 80E70F         <1>     and     bh, 00001111b           ; KEEP ONLY MOTOR BITS
00004CA3 8025[06770100]CF   <1>     and     byte [MOTOR_STATUS], 11001111b ; CLEAR OUT DRIVE SELECT
00004CAA 0825[06770100]     <1>     or      [MOTOR_STATUS], ah      ; OR IN DRIVE SELECTED AND MOTOR ON
00004CB0 A0[06770100]       <1>     mov     al, [MOTOR_STATUS]      ; GET DIGITAL OUTPUT REGISTER REFLECTION
00004CB5 88C3           <1>     mov     bl, al                  ; BL=@MOTOR_STATUS AFTER, BH=BEFORE
00004CB7 80E30F         <1>     and     bl, 00001111b           ; KEEP ONLY MOTOR BITS
00004CBA FB             <1>     sti                             ; ENABLE INTERRUPTS AGAIN
00004CBB 243F           <1>     and     al, 00111111b           ; STRIP AWAY UNWANTED BITS
00004CBD C0C004         <1>     rol     al, 4                   ; PUT BITS IN DESIRED POSITIONS
00004CC0 0C0C           <1>     or      al, 00001100b           ; NO RESET, ENABLE DMA/INTERRUPT
00004CC2 66BAF203       <1>     mov     dx, 03F2h               ; SELECT DRIVE AND TURN ON MOTOR
00004CC6 EE             <1>     out     dx, al
00004CC7 38FB           <1>     cmp     bl, bh                  ; NEW MOTOR TURNED ON ?
                        <1>     ;jz     short NO_MOT_WAIT       ; NO WAIT REQUIRED IF JUST SELECT
00004CC9 7403           <1>     je      short no_mot_w1 ; 27/02/2015
00004CCB F8             <1>     clc                             ; (re)SET CARRY MEANING WAIT
00004CCC C3             <1>     retn
                        <1>
                        <1> NO_MOT_WAIT:
00004CCD FB             <1>     sti
                        <1> no_mot_w1: ; 27/02/2015
00004CCE F9             <1>     stc                             ; SET NO WAIT REQUIRED
                        <1>     ;sti                            ; INTERRUPTS BACK ON
00004CCF 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:
                        <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
                        <1>     ;mov    dl,9                    ; GET HEAD SETTLE PARAMETER
                        <1>     ; 06/08/2022
00004CD0 B009           <1>     mov     al, 9
00004CD2 E835FFFFFF     <1>     call    GET_PARM
00004CD7 08E4           <1>     or      ah, ah  ; 17/12/2014
00004CD9 7519           <1>     jnz     short DO_WAT
00004CDB F605[06770100]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
00004CE2 741D           <1>     jz      short HW_DONE
00004CE4 B40F           <1>     mov     ah, HD12_SETTLE                 ; LOAD 1.2M HEAD SETTLE MINIMUM
00004CE6 8A87[13770100]     <1>     mov     al, [DSK_STATE+edi]     ; LOAD STATE
00004CEC 24C0           <1>     and     al, RATE_MSK            ; KEEP ONLY RATE
00004CEE 3C80           <1>     cmp     al, RATE_250            ; 1.2 M DRIVE ?
00004CF0 7502           <1>     jnz     short DO_WAT            ; DEFAULT HEAD SETTLE LOADED
                        <1> ;GP3:
00004CF2 B414           <1>     mov     ah, HD320_SETTLE        ; USE 320/360 HEAD SETTLE
                        <1> ;   jmp     short DO_WAT
                        <1>
                        <1> ;ISNT_WRITE:
                        <1> ;   or      ah, ah                  ; CHECK FOR NO WAIT
                        <1> ;   jz      short HW_DONE           ; IF NOT WRITE AND 0 ITS OK
                        <1>
                        <1> ;-----      AH CONTAINS NUMBER OF MILLISECONDS TO WAIT
                        <1> DO_WAT:
                        <1> ;   mov     al, ah                  ; AL = # MILLISECONDS
                        <1> ;   ;xor    ah, ah                  ; AX = # MILLISECONDS
```

```
                        <1> J29:                                ;       1 MILLISECOND LOOP
                        <1>     ;;mov   cx, WAIT_FDU_HEAD_SETTLE ; 33 ; 1 ms in 30 micro units.
                        <1>     ;mov    ecx, 66                 ; COUNT AT 15.085737 US PER COUNT
                        <1>     ; 11/04/2021
                        <1>     ;mov    ecx, WAIT_FDU_HEAD_SETTLE ; 33
                        <1>     ; 06/08/2022
00004CF4 29C9           <1>     sub     ecx, ecx
00004CF6 B121           <1>     mov     cl, WAIT_FDU_HEAD_SETTLE ; 33
00004CF8 E824D7FFFF     <1>     call    WAITF                   ; DELAY FOR 1 MILLISECOND
                        <1>     ;dec    al                      ; DECREMENT THE COUNT
00004CFD FECC           <1>     dec     ah
00004CFF 75F3           <1>     jnz     short J29              ; DO AL MILLISECOND # OF TIMES
                        <1> HW_DONE:
00004D01 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>     ; 09/08/2022
                        <1>     ; 06/08/2022 - TRDOS 386 Kernel v2.0.5
                        <1>     ;
                        <1>     ;push   bx                      ; SAVE REG.
00004D02 66BAF403       <1>     mov     dx, 03F4h               ; STATUS PORT
                        <1>     ;mov    bl,2                    ; HIGH ORDER COUNTER
                        <1>     ;xor    cx, cx                  ; COUNT FOR TIME OUT
                        <1>     ; 16/12/2014
                        <1>     ; waiting for (max.) 0.5 seconds
                        <1>       ;;mov      byte [wait_count], 0 ;; 27/02/2015
                        <1>     ;
                        <1>     ; 17/12/2014
                        <1>     ; Modified from AWARD BIOS 1999 - ADISK.ASM - SEND_COMMAND
                        <1>     ;
                        <1>     ;WAIT_FOR_PORT:       Waits for a bit at a port pointed to by DX to
                        <1>     ;                     go on.
                        <1>     ;INPUT:
                        <1>     ;       AH=Mask for isolation bits.
                        <1>     ;       AL=pattern to look for.
                        <1>     ;       DX=Port to test for
                        <1>     ;       BH:CX=Number of memory refresh periods to delay.
                        <1>     ;            (normally 30 microseconds per period.)
                        <1>     ;
                        <1>     ;WFP_SHORT:
                        <1>     ;       Wait for port if refresh cycle is short (15-80 Us range).
                        <1>     ;
                        <1>
                        <1> ;   mov     bl, WAIT_FDU_SEND_HI+1; 0+1
                        <1> ;   mov     cx, WAIT_FDU_SEND_LO  ; 16667
00004D06 B91B410000     <1>     mov     ecx, WAIT_FDU_SEND_LH   ; 16667 (27/02/2015)
                        <1> ;
                        <1> ;WFPS_OUTER_LP:
                        <1> ;   ;
                        <1> ;WFPS_CHECK_PORT:
                        <1>
00004D0B EC             <1> J23:    in      al, dx                  ; GET STATUS
00004D0C 24C0           <1>     and     al, 11000000b           ; KEEP STATUS AND DIRECTION
00004D0E 3C80           <1>     cmp     al, 10000000b           ; STATUS 1 AND DIRECTION 0 ?
00004D10 7418           <1>     jz      short J27               ; STATUS AND DIRECTION OK
                        <1> WFPS_HI:
00004D12 E461           <1>     in      al, PORT_B  ; 061h      ; SYS1  ; wait for hi to lo
00004D14 A810           <1>     test    al, 010h                ; transition on memory
00004D16 75FA           <1>     jnz     short WFPS_HI           ; refresh.
                        <1> WFPS_LO:
00004D18 E461           <1>     in      al, PORT_B              ; SYS1
00004D1A A810           <1>     test    al, 010h
00004D1C 74FA           <1>     jz      short WFPS_LO
                        <1>     ;loop   short WFPS_CHECK_PORT
00004D1E E2EB           <1>     loop    J23     ; 27/02/2015
                        <1> ;   ;
                        <1> ;   dec     bl
                        <1> ;   jnz     short WFPS_OUTER_LP
                        <1> ;   jmp     short WFPS_TIMEOUT      ; fail
                        <1> ;J23:
                        <1> ;   in      al, dx                  ; GET STATUS
                        <1> ;   and     al, 11000000b           ; KEEP STATUS AND DIRECTION
                        <1> ;   cmp     al, 10000000b           ; STATUS 1 AND DIRECTION 0 ?
                        <1> ;   jz      short J27               ; STATUS AND DIRECTION OK
                        <1>     ;loop   J23                     ; CONTINUE TILL CX EXHAUSTED
                        <1>     ;dec    bl                      ; DECREMENT COUNTER
                        <1>     ;jnz    short J23               ; REPEAT TILL DELAY FINISHED, CX = 0
                        <1>
                        <1>     ;;27/02/2015
                        <1>     ;16/12/2014
                        <1>         ;;cmp      byte [wait_count], 10   ; (10/18.2 seconds)
                        <1>     ;;jb    short J23
                        <1>
                        <1> ;WFPS_TIMEOUT:
                        <1>
                        <1> ;-----      FALL THRU TO ERROR RETURN
                        <1>
```

```
3202 00004D20 800D[08770100]80    <1>     or      byte [DSKETTE_STATUS], TIME_OUT
3203                              <1>     ;pop    bx                      ; RESTORE REG.
3204 00004D27 58                  <1>     pop     eax ; 08/02/2015        ; DISCARD THE RETURN ADDRESS
3205 00004D28 F9                  <1>     stc                             ; INDICATE ERROR TO CALLER
3206 00004D29 C3                  <1>     retn
3207                              <1>
3208                              <1> ;-----      DIRECTION AND STATUS OK; OUTPUT BYTE
3209                              <1>
3210                              <1> J27:
3211 00004D2A 88E0                <1>     mov     al, ah                  ; GET BYTE TO OUTPUT
3212                              <1>     ;inc    dx                      ; DATA PORT = STATUS PORT + 1
3213                              <1>     ; 06/08/2022
3214 00004D2C FEC2                <1>     inc     dl
3215 00004D2E EE                  <1>     out     dx, al                  ; OUTPUT THE BYTE
3216                              <1>     ;;NEWIODELAY  ;; 27/02/2015
3217                              <1>     ; 27/02/2015
3218                              <1>     ;pushf                          ; SAVE FLAGS
3219                              <1>     ; 09/08/2022
3220                              <1>     ; cf = 0, zf = 1
3221                              <1>     ;mov    ecx, 3                  ; 30 TO 45 MICROSECONDS WAIT FOR
3222                              <1>     ; 11/04/2021
3223                              <1>     ;mov    ecx, 2
3224                              <1>     ; 06/08/2022
3225 00004D2F 29C9                <1>     sub     ecx, ecx
3226 00004D31 B102                <1>     mov     cl, 2
3227 00004D33 E8E9D6FFFF          <1>     call    WAITF                   ; NEC FLAGS UPDATE CYCLE
3228                              <1>     ; 09/08/2022
3229                              <1>     ; cf = 0, zf = 1
3230                              <1>     ;popf                           ; RESTORE FLAGS FOR EXIT
3231                              <1>     ;pop    bx                      ; RESTORE REG
3232 00004D38 C3                  <1>     retn                            ; CY = 0 FROM TEST INSTRUCTION
3233                              <1>
3234                              <1> ;---------------------------------------------------------------------------
3235                              <1> ; SEEK
3236                              <1> ;   THIS ROUTINE WILL MOVE THE HEAD ON THE NAMED DRIVE TO THE NAMED
3237                              <1> ;   TRACK. IF THE DRIVE HAS NOT BEEN ACCESSED SINCE THE DRIVE
3238                              <1> ;   RESET COMMAND WAS ISSUED, THE DRIVE WILL BE RECALIBRATED.
3239                              <1> ;
3240                              <1> ; ON ENTRY: DI = DRIVE #
3241                              <1> ;           CH = TRACK #
3242                              <1> ;
3243                              <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
3244                              <1> ;           AX,BX,CX DX DESTROYED
3245                              <1> ;---------------------------------------------------------------------------
3246                              <1> SEEK:
3247 00004D39 89FB                <1>     mov     ebx, edi                ; BX = DRIVE #
3248 00004D3B B001                <1>     mov     al, 1                   ; ESTABLISH MASK FOR RECALIBRATE TEST
3249                              <1>     ; 06/08/2022
3250                              <1>     ;xchg   cl, bl                  ; SET DRIVE VALUE INTO CL
3251                              <1>     ;rol    al, cl                  ; SHIFT MASK BY THE DRIVE VALUE
3252                              <1>     ;xchg   cl, bl                  ; RECOVER TRACK VALUE
3253                              <1>     ; 06/08/2022
3254 00004D3D 84C3                <1>     test    bl, al ; test bl, 1
3255 00004D3F 7402                <1>     jz      short seek_0
3256 00004D41 FEC0                <1>     inc     al ; shl al, 1
3257                              <1> seek_0:
3258 00004D43 8405[05770100]      <1>     test    al, [SEEK_STATUS]       ; TEST FOR RECALIBRATE REQUIRED
3259 00004D49 7526                <1>     jnz     short J28A              ; JUMP IF RECALIBRATE NOT REQUIRED
3260                              <1>
3261 00004D4B 0805[05770100]      <1>     or      [SEEK_STATUS], al       ; TURN ON THE NO RECALIBRATE BIT IN FLAG
3262 00004D51 E862000000          <1>     call    RECAL                   ; RECALIBRATE DRIVE
3263 00004D56 730E                <1>     jnc     short AFT_RECAL                 ; RECALIBRATE DONE
3264                              <1>
3265                              <1> ;-----      ISSUE RECALIBRATE FOR 80 TRACK DISKETTES
3266                              <1>
3267 00004D58 C605[08770100]00    <1>     mov     byte [DSKETTE_STATUS], 0 ; CLEAR OUT INVALID STATUS
3268 00004D5F E854000000          <1>     call    RECAL                   ; RECALIBRATE DRIVE
3269 00004D64 7251                <1>     jc      short RB                ; IF RECALIBRATE FAILS TWICE THEN ERROR
3270                              <1>
3271                              <1> AFT_RECAL:
3272 00004D66 C687[15770100]00    <1>         mov     byte [DSK_TRK+edi], 0    ; SAVE NEW CYLINDER AS PRESENT POSITION
3273 00004D6D 08ED                <1>     or      ch, ch                  ; CHECK FOR SEEK TO TRACK 0
3274 00004D6F 743F                <1>     jz      short DO_WAIT           ; HEAD SETTLE, CY = 0 IF JUMP
3275                              <1>
3276                              <1> ;-----      DRIVE IS IN SYNCHRONIZATION WITH CONTROLLER, SEEK TO TRACK
3277                              <1>
3278 00004D71 F687[13770100]20    <1> J28A:       test    byte [DSK_STATE+edi], DBL_STEP ; CHECK FOR DOUBLE STEP REQUIRED
3279 00004D78 7402                <1>     jz      short _R7               ; SINGLE STEP REQUIRED BYPASS DOUBLE
3280 00004D7A D0E5                <1>     shl     ch, 1                   ; DOUBLE NUMBER OF STEP TO TAKE
3281                              <1>
3282 00004D7C 3AAF[15770100]      <1> _R7:cmp     ch, [DSK_TRK+edi]       ; SEE IF ALREADY AT THE DESIRED TRACK
3283 00004D82 7433                <1>     je      short RB                ; IF YES, DO NOT NEED TO SEEK
3284                              <1>
3285 00004D84 BA[B74D0000]        <1>     mov     edx, NEC_ERR            ; LOAD RETURN ADDRESS
3286 00004D89 52                  <1>     push    edx ; (*)               ; ON STACK FOR NEC OUTPUT ERROR
3287 00004D8A 88AF[15770100]      <1>     mov     [DSK_TRK+edi], ch       ; SAVE NEW CYLINDER AS PRESENT POSITION
3288 00004D90 B40F                <1>     mov     ah, 0Fh                 ; SEEK COMMAND TO NEC
3289 00004D92 E86BFFFFFF          <1>     call    NEC_OUTPUT
3290 00004D97 89FB                <1>     mov     ebx, edi                ; BX = DRIVE #
3291 00004D99 88DC                <1>     mov     ah, bl                  ; OUTPUT DRIVE NUMBER
3292 00004D9B E862FFFFFF          <1>     call    NEC_OUTPUT
3293 00004DA0 8AA7[15770100]      <1>     mov     ah, [DSK_TRK+edi]       ; GET CYLINDER NUMBER
3294 00004DA6 E857FFFFFF          <1>     call    NEC_OUTPUT
3295 00004DAB E827000000          <1>     call    CHK_STAT_2              ; ENDING INTERRUPT AND SENSE STATUS
3296                              <1>
3297                              <1> ;-----      WAIT FOR HEAD SETTLE
3298                              <1>
3299                              <1> DO_WAIT:
3300 00004DB0 9C                  <1>     pushf                           ; SAVE STATUS
3301 00004DB1 E81AFFFFFF          <1>     call    HD_WAIT                 ; WAIT FOR HEAD SETTLE TIME
3302 00004DB6 9D                  <1>     popf                            ; RESTORE STATUS
3303                              <1> RB:
3304                              <1> NEC_ERR:
3305                              <1>     ; 08/02/2015 (code trick here from original IBM PC/AT DISKETTE.ASM)
3306                              <1>     ; (*) nec_err -> retn (push edx -> pop edx) -> nec_err -> retn
3307 00004DB7 C3                  <1>     retn                            ; RETURN TO CALLER
3308                              <1>
3309                              <1> ;---------------------------------------------------------------------------
3310                              <1> ; RECAL
3311                              <1> ;   RECALIBRATE DRIVE
3312                              <1> ;
3313                              <1> ; ON ENTRY: DI = DRIVE #
3314                              <1> ;
3315                              <1> ; ON EXIT:  CY REFLECTS STATUS OF OPERATION.
3316                              <1> ;---------------------------------------------------------------------------
3317                              <1> RECAL:
3318                              <1>     ;push   cx
3319                              <1>     ; 11/04/2021
3320 00004DB8 51                  <1>     push    ecx
3321 00004DB9 B8[D54D0000]        <1>     mov     eax, RC_BACK            ; LOAD NEC_OUTPUT ERROR
3322 00004DBE 50                  <1>     push    eax
3323 00004DBF B407                <1>     mov     ah, 07h                 ; RECALIBRATE COMMAND
3324 00004DC1 E83CFFFFFF          <1>     call    NEC_OUTPUT
3325 00004DC6 89FB                <1>     mov     ebx, edi                ; BX = DRIVE #
```

```
3326 00004DC8 88DC              <1>     mov     ah, bl
3327 00004DCA E833FFFFFF        <1>     call    NEC_OUTPUT              ; OUTPUT THE DRIVE NUMBER
3328 00004DCF E803000000        <1>     call    CHK_STAT_2              ; GET THE INTERRUPT AND SENSE INT STATUS
3329 00004DD4 58                <1>     pop     eax                     ; THROW AWAY ERROR
3330                            <1> RC_BACK:
3331                            <1>     ;pop    cx
3332                            <1>     ; 11/04/2021
3333 00004DD5 59                <1>     pop     ecx
3334 00004DD6 C3                <1>     retn
3335                            <1>
3336                            <1> ;--------------------------------------------------------------------------------
3337                            <1> ; CHK_STAT_2
3338                            <1> ;   THIS ROUTINE HANDLES THE INTERRUPT RECEIVED AFTER RECALIBRATE,
3339                            <1> ;   OR SEEK TO THE ADAPTER. THE INTERRUPT IS WAITED FOR, THE
3340                            <1> ;   INTERRUPT STATUS SENSED, AND THE RESULT RETURNED TO THE CALLER.
3341                            <1> ;
3342                            <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
3343                            <1> ;--------------------------------------------------------------------------------
3344                            <1> CHK_STAT_2:
3345 00004DD7 B8[FF4D0000]      <1>         mov     eax, CS_BACK            ; LOAD NEC_OUTPUT ERROR ADDRESS
3346 00004DDC 50                <1>     push    eax
3347 00004DDD E828000000        <1>     call    WAIT_INT                ; WAIT FOR THE INTERRUPT
3348 00004DE2 721A              <1>     jc      short J34               ; IF ERROR, RETURN IT
3349 00004DE4 B408              <1>     mov     ah, 08h                 ; SENSE INTERRUPT STATUS COMMAND
3350 00004DE6 E817FFFFFF        <1>     call    NEC_OUTPUT
3351 00004DEB E849000000        <1>     call    RESULTS                 ; READ IN THE RESULTS
3352 00004DF0 720C              <1>     jc      short J34
3353 00004DF2 A0[09770100]      <1>     mov     al, [NEC_STATUS]        ; GET THE FIRST STATUS BYTE
3354 00004DF7 2460              <1>     and     al, 01100000B           ; ISOLATE THE BITS
3355 00004DF9 3C60              <1>     cmp     al, 01100000B           ; TEST FOR CORRECT VALUE
3356 00004DFB 7403              <1>     jz      short J35               ; IF ERROR, GO MARK IT
3357 00004DFD F8                <1>     clc                             ; GOOD RETURN
3358                            <1> J34:
3359 00004DFE 58                <1>     pop     eax                     ; THROW AWAY ERROR RETURN
3360                            <1> CS_BACK:
3361 00004DFF C3                <1>     retn
3362                            <1> J35:
3363 00004E00 800D[08770100]40  <1>     or      byte [DSKETTE_STATUS], BAD_SEEK
3364 00004E07 F9                <1>     stc                             ; ERROR RETURN CODE
3365 00004E08 EBF4              <1>     jmp     short J34
3366                            <1>
3367                            <1> ;--------------------------------------------------------------------------------
3368                            <1> ; WAIT_INT
3369                            <1> ;   THIS ROUTINE WAITS FOR AN INTERRUPT TO OCCUR A TIME OUT ROUTINE
3370                            <1> ;   TAKES PLACE DURING THE WAIT, SO THAT AN ERROR MAY BE RETURNED
3371                            <1> ;   IF THE DRIVE IS NOT READY.
3372                            <1> ;
3373                            <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
3374                            <1> ;--------------------------------------------------------------------------------
3375                            <1>
3376                            <1> ; 17/12/2014
3377                            <1> ; 2.5 seconds waiting !
3378                            <1> ;(AWARD BIOS - 1999, WAIT_FDU_INT_LOW, WAIT_FDU_INT_HI)
3379                            <1> ; amount of time to wait for completion interrupt from NEC.
3380                            <1>
3381                            <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
3382                            <1> WAIT_INT:
3383 00004E0A FB                <1>     sti                             ; TURN ON INTERRUPTS, JUST IN CASE
3384                            <1>     ; 06/08/2022
3385                            <1>     ;clc                            ; CLEAR TIMEOUT INDICATOR
3386                            <1>     ;
3387                            <1>     ;mov    bl, 10                  ; CLEAR THE COUNTERS
3388                            <1>     ;xor    cx, cx                  ; FOR 2 SECOND WAIT
3389                            <1>
3390                            <1>     ; Modification from AWARD BIOS - 1999 (ATORGS.ASM, WAIT
3391                            <1>     ;
3392                            <1>     ;WAIT_FOR_MEM:
3393                            <1>     ;       Waits for a bit at a specified memory location pointed
3394                            <1>     ;       to by ES:[DI] to become set.
3395                            <1>     ;INPUT:
3396                            <1>     ;       AH= Mask to test with.
3397                            <1>     ;       ES:[DI] = memory location to watch.
3398                            <1>     ;       BH:CX= Number of memory refresh periods to delay.
3399                            <1>     ;            (normally 30 microseconds per period.)
3400                            <1>
3401                            <1>     ; waiting for (max.) 2.5 secs in 30 micro units.
3402                            <1> ;   mov     cx, WAIT_FDU_INT_LO             ; 017798
3403                            <1> ;;  mov     bl, WAIT_FDU_INT_HI
3404                            <1> ;   mov     bl, WAIT_FDU_INT_HI + 1
3405                            <1>     ; 27/02/2015
3406 00004E0B B986450100        <1>     mov     ecx, WAIT_FDU_INT_LH    ; 83334 (2.5 seconds)
3407                            <1> WFMS_CHECK_MEM:
3408 00004E10 F605[05770100]80  <1>     test    byte [SEEK_STATUS], INT_FLAG
3409                            <1>                                     ; TEST FOR INTERRUPT OCCURRING
3410 00004E17 7516              <1>         jnz     short J37
3411                            <1> WFMS_HI:
3412 00004E19 E461              <1>     in      al, PORT_B  ; 061h      ; SYS1, wait for lo to hi
3413 00004E1B A810              <1>     test    al, 010h                ; transition on memory
3414 00004E1D 75FA              <1>     jnz     short WFMS_HI           ; refresh.
3415                            <1> WFMS_LO:
3416 00004E1F E461              <1>     in      al, PORT_B              ; SYS1
3417 00004E21 A810              <1>     test    al, 010h
3418 00004E23 74FA              <1>     jz      short WFMS_LO
3419 00004E25 E2E9              <1>         loop        WFMS_CHECK_MEM
3420                            <1>
3421                            <1> ;WFMS_OUTER_LP:
3422                            <1> ;;  or      bl, bl                  ; check outer counter
3423                            <1> ;;  jz      short J36A              ; WFMS_TIMEOUT
3424                            <1> ;   dec     bl
3425                            <1> ;   jz      short J36A
3426                            <1> ;   jmp     short WFMS_CHECK_MEM
3427                            <1>
3428                            <1>     ; 17/12/2014
3429                            <1>     ; 16/12/2014
3430                            <1> ;   mov     byte [wait_count], 0    ; Reset (INT 08H) counter
3431                            <1> ;J36:
3432                            <1> ;   test    byte [SEEK_STATUS], INT_FLAG
3433                            <1> ;                                   ; TEST FOR INTERRUPT OCCURRING
3434                            <1> ;   jnz     short J37
3435                            <1>
3436                            <1>     ; 16/12/2014
3437                            <1>     ;loop   J36                     ; COUNT DOWN WHILE WAITING
3438                            <1>     ;dec    bl                      ; SECOND LEVEL COUNTER
3439                            <1>     ;jnz    short J36
3440                            <1> ;       cmp     byte [wait_count], 46   ; (46/18.2 seconds)
3441                            <1> ;   jb      short J36
3442                            <1>
3443                            <1> ;WFMS_TIMEOUT:
3444                            <1> ;J36A:
3445 00004E27 800D[08770100]80  <1>     or      byte [DSKETTE_STATUS], TIME_OUT ; NOTHING HAPPENED
3446 00004E2E F9                <1>     stc                             ; ERROR RETURN
3447                            <1> J37:
3448 00004E2F 9C                <1>     pushf                           ; SAVE CURRENT CARRY
3449 00004E30 8025[05770100]7F  <1>     and     byte [SEEK_STATUS], ~INT_FLAG ; TURN OFF INTERRUPT FLAG
```

```
3450 00004E37 9D                <1>     popf                            ; RECOVER CARRY
3451 00004E38 C3                <1>     retn                            ; GOOD RETURN CODE
3452                            <1>
3453                            <1> ;--------------------------------------------------------------------------------
3454                            <1> ; RESULTS
3455                            <1> ;   THIS ROUTINE WILL READ ANYTHING THAT THE NEC CONTROLLER RETURNS
3456                            <1> ;   FOLLOWING AN INTERRUPT.
3457                            <1> ;
3458                            <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
3459                            <1> ;           AX,BX,CX,DX DESTROYED
3460                            <1> ;--------------------------------------------------------------------------------
3461                            <1> RESULTS:
3462                            <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
3463 00004E39 57               <1>     push    edi
3464 00004E3A BF[09770100]      <1>     mov     edi, NEC_STATUS                 ; POINTER TO DATA AREA
3465 00004E3F B307              <1>     mov     bl, 7                   ; MAX STATUS BYTES
3466 00004E41 66BAF403          <1>     mov     dx, 03F4h               ; STATUS PORT
3467                            <1>
3468                            <1> ;-----      WAIT FOR REQUEST FOR MASTER
3469                            <1>
3470                            <1> _R10:
3471                            <1>     ; 16/12/2014
3472                            <1>     ; wait for (max) 0.5 seconds
3473                            <1>     ;mov    bh, 2                   ; HIGH ORDER COUNTER
3474                            <1>     ;xor    cx, cx                  ; COUNTER
3475                            <1>
3476                            <1>     ;Time to wait while waiting for each byte of NEC results = .5
3477                            <1>     ;seconds.  .5 seconds = 500,000 micros.  500,000/30 = 16,667.
3478                            <1>     ; 27/02/2015
3479 00004E45 B91B410000        <1>     mov     ecx, WAIT_FDU_RESULTS_LH ; 16667
3480                            <1>     ;mov    cx, WAIT_FDU_RESULTS_LO  ; 16667
3481                            <1>     ;mov    bh, WAIT_FDU_RESULTS_HI+1 ; 0+1
3482                            <1>
3483                            <1> WFPSR_OUTER_LP:
3484                            <1>     ;
3485                            <1> WFPSR_CHECK_PORT:
3486                            <1> J39:                                ; WAIT FOR MASTER
3487 00004E4A EC                <1>     in      al, dx                  ; GET STATUS
3488 00004E4B 24C0              <1>     and     al, 11000000b           ; KEEP ONLY STATUS AND DIRECTION
3489 00004E4D 3CC0              <1>     cmp     al, 11000000b           ; STATUS 1 AND DIRECTION 1 ?
3490 00004E4F 7418              <1>     jz      short J42               ; STATUS AND DIRECTION OK
3491                            <1> WFPSR_HI:
3492 00004E51 E461              <1>     in      al, PORT_B      ; 061h  ; SYS1  ; wait for hi to lo
3493 00004E53 A810              <1>     test    al, 010h                ; transition on memory
3494 00004E55 75FA              <1>     jnz     short WFPSR_HI          ; refresh.
3495                            <1> WFPSR_LO:
3496 00004E57 E461              <1>     in      al, PORT_B              ; SYS1
3497 00004E59 A810              <1>     test    al, 010h
3498 00004E5B 74FA              <1>     jz      SHORT WFPSR_LO
3499 00004E5D E2EB              <1>     loop    WFPSR_CHECK_PORT
3500                            <1>
3501                            <1>     ;; 27/02/2015
3502                            <1>     ;;dec   bh
3503                            <1>     ;;jnz   short WFPSR_OUTER_LP
3504                            <1>     ;jmp    short WFPSR_TIMEOUT     ; fail
3505                            <1>
3506                            <1>     ;;mov   byte [wait_count], 0
3507                            <1> ;J39:                                       ; WAIT FOR MASTER
3508                            <1> ;   in      al, dx                  ; GET STATUS
3509                            <1> ;   and     al, 11000000b           ; KEEP ONLY STATUS AND DIRECTION
3510                            <1> ;   cmp     al, 11000000b           ; STATUS 1 AND DIRECTION 1 ?
3511                            <1> ;   jz      short J42               ; STATUS AND DIRECTION OK
3512                            <1>     ;loop   J39                     ; LOOP TILL TIMEOUT
3513                            <1>     ;dec    bh                      ; DECREMENT HIGH ORDER COUNTER
3514                            <1>     ;jnz    short J39               ; REPEAT TILL DELAY DONE
3515                            <1>     ;
3516                            <1>     ;;cmp   byte [wait_count], 10   ; (10/18.2 seconds)
3517                            <1>     ;;jb    short J39
3518                            <1>
3519                            <1> ;WFPSR_TIMEOUT:
3520 00004E5F 800D[08770100]80  <1>     or      byte [DSKETTE_STATUS], TIME_OUT
3521 00004E66 F9                <1>     stc                             ; SET ERROR RETURN
3522 00004E67 EB28              <1>     jmp     short POPRES            ; POP REGISTERS AND RETURN
3523                            <1>
3524                            <1> ;-----      READ IN THE STATUS
3525                            <1>
3526                            <1> J42:
3527 00004E69 EB00              <1>     jmp     $+2                     ; I/O DELAY
3528                            <1>     ;inc    dx                      ; POINT AT DATA PORT
3529                            <1>     ; 06/08/2022
3530 00004E6B FEC2              <1>     inc     dl
3531 00004E6D EC                <1>     in      al, dx                  ; GET THE DATA
3532                            <1>     ; 16/12/2014
3533                            <1>     NEWIODELAY
  83 00004E6E E6EB              <2>  out 0EBh, al
3534 00004E70 8807              <1>         mov     [edi], al           ; STORE THE BYTE
3535 00004E72 47                <1>     inc     edi                     ; INCREMENT THE POINTER
3536                            <1>
3537                            <1>     ; 16/12/2014
3538                            <1> ;   push    cx
3539                            <1> ;   mov     cx, 30
3540                            <1> ;wdw2:
3541                            <1> ;   NEWIODELAY
3542                            <1> ;   loop    wdw2
3543                            <1> ;   pop     cx
3544                            <1>
3545                            <1>     ;;mov   ecx, 3                  ; MINIMUM 24 MICROSECONDS FOR NEC
3546                            <1>     ; 11/04/2021
3547                            <1>     ;mov    ecx, 2
3548                            <1>     ; 06/08/2022
3549 00004E73 29C9              <1>     sub     ecx, ecx
3550 00004E75 B102              <1>     mov     cl, 2
3551 00004E77 E8A5D5FFFF        <1>     call    WAITF                   ; WAIT 30 TO 45 MICROSECONDS
3552                            <1>     ;dec    dx                      ; POINT AT STATUS PORT
3553                            <1>     ; 06/08/2022
3554 00004E7C FECA              <1>     dec     dl
3555 00004E7E EC                <1>     in      al, dx                  ; GET STATUS
3556                            <1>     ; 16/12/2014
3557                            <1>     NEWIODELAY
  83 00004E7F E6EB              <2>  out 0EBh, al
3558                            <1>     ;
3559 00004E81 A810              <1>     test    al, 00010000b           ; TEST FOR NEC STILL BUSY
3560 00004E83 740C              <1>     jz      short POPRES            ; RESULTS DONE ?
3561                            <1>
3562 00004E85 FECB              <1>     dec     bl                      ; DECREMENT THE STATUS COUNTER
3563 00004E87 75BC              <1>     jnz     short _R10              ; GO BACK FOR MORE
3564 00004E89 800D[08770100]20  <1>     or      byte [DSKETTE_STATUS], BAD_NEC ; TOO MANY STATUS BYTES
3565 00004E90 F9                <1>     stc                             ; SET ERROR FLAG
3566                            <1>
3567                            <1> ;-----      RESULT OPERATION IS DONE
3568                            <1> POPRES:
3569 00004E91 5F                <1>     pop     edi
3570 00004E92 C3                <1>     retn                            ; RETURN WITH CARRY SET
3571                            <1>
```

```
3572                            <1> ;--------------------------------------------------------------------------------
3573                            <1> ; READ_DSKCHNG
3574                            <1> ;   READS THE STATE OF THE DISK CHANGE LINE.
3575                            <1> ;
3576                            <1> ; ON ENTRY: DI = DRIVE #
3577                            <1> ;
3578                            <1> ; ON EXIT:  DI = DRIVE #
3579                            <1> ;           ZF = 0 : DISK CHANGE LINE INACTIVE
3580                            <1> ;           ZF = 1 : DISK CHANGE LINE ACTIVE
3581                            <1> ;           AX,CX,DX DESTROYED
3582                            <1> ;--------------------------------------------------------------------------------
3583                            <1> READ_DSKCHNG:
3584 00004E93 E8AFFDFFFF       <1>     call    MOTOR_ON                ; TURN ON THE MOTOR IF OFF
3585 00004E98 66BAF703         <1>     mov     dx, 03F7h               ; ADDRESS DIGITAL INPUT REGISTER
3586 00004E9C EC               <1>     in      al, dx                  ; INPUT DIGITAL INPUT REGISTER
3587 00004E9D A880             <1>     test    al, DSK_CHG             ; CHECK FOR DISK CHANGE LINE ACTIVE
3588 00004E9F C3               <1>     retn                            ; RETURN TO CALLER WITH ZERO FLAG SET
3589                            <1>
3590                            <1> fdc_int:
3591                            <1>       ; 30/07/2015
3592                            <1>       ; 16/02/2015
3593                            <1> ;int_0Eh: ; 11/12/2014
3594                            <1>
3595                            <1> ;--- HARDWARE INT 0EH -- ( IRQ LEVEL 6 ) ------------------------------------------
3596                            <1> ; DISK_INT
3597                            <1> ;   THIS ROUTINE HANDLES THE DISKETTE INTERRUPT.
3598                            <1> ;
3599                            <1> ; ON EXIT:  THE INTERRUPT FLAG IS SET IN @SEEK_STATUS.
3600                            <1> ;--------------------------------------------------------------------------------
3601                            <1> DISK_INT_1:
3602                            <1>     ;push   AX                      ; SAVE WORK REGISTER
3603                            <1>     ; 11/04/2021
3604 00004EA0 50               <1>     push    eax
3605 00004EA1 1E               <1>     push    ds
3606 00004EA2 66B81000         <1>     mov     ax, KDATA
3607 00004EA6 8ED8             <1>     mov     ds, ax
3608 00004EA8 800D[05770100]80 <1>         or  byte [SEEK_STATUS], INT_FLAG ; TURN ON INTERRUPT OCCURRED
3609 00004EAF B020             <1>     mov     al, EOI                 ; END OF INTERRUPT MARKER
3610 00004EB1 E620             <1>     out     INTA00, al              ; INTERRUPT CONTROL PORT
3611 00004EB3 1F               <1>     pop     ds
3612                            <1>     ;pop    ax                      ; RECOVER REGISTER
3613                            <1>     ; 11/04/2021
3614 00004EB4 58               <1>     pop     eax
3615 00004EB5 CF               <1>     iretd                           ; RETURN FROM INTERRUPT
3616                            <1>
3617                            <1> ;--------------------------------------------------------------------------------
3618                            <1> ; DSKETTE_SETUP
3619                            <1> ;   THIS ROUTINE DOES A PRELIMINARY CHECK TO SEE WHAT TYPE OF
3620                            <1> ;   DISKETTE DRIVES ARE ATTACH TO THE SYSTEM.
3621                            <1> ;--------------------------------------------------------------------------------
3622                            <1>
3623                            <1> ; 09/08/2022 - TRDOS 386 Kernel v2.0.5
3624                            <1> ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
3625                            <1>
3626                            <1> DSKETTE_SETUP:
3627                            <1>     ;push   ax                      ; SAVE REGISTERS
3628                            <1>     ;push   bx
3629                            <1>     ;push   cx
3630 00004EB6 52               <1>     push    edx
3631                            <1>     ;push   di
3632                            <1>     ;;push  ds
3633                            <1>     ; 14/12/2014
3634                            <1>     ;mov    word [DISK_POINTER], MD_TBL6
3635                            <1>     ;mov    [DISK_POINTER+2], cs
3636                            <1>     ;
3637                            <1>     ;or     byte [RTC_WAIT_FLAG], 1       ; NO RTC WAIT, FORCE USE OF LOOP
3638 00004EB7 31FF             <1>     xor     edi, edi                ; INITIALIZE DRIVE POINTER
3639                            <1>     ; 09/08/2022
3640                            <1>     ;mov    esi, eax
3641 00004EB9 31C0             <1>     xor     eax, eax ; eax = 0
3642 00004EBB 66A3[13770100]   <1>     mov     [DSK_STATE], ax             ; INITIALIZE STATES
3643 00004EC1 8025[10770100]33 <1>     and     byte [LASTRATE], ~(STRT_MSK+SEND_MSK) ; CLEAR START & SEND
3644 00004EC8 800D[10770100]C0 <1>     or      byte [LASTRATE], SEND_MSK ; INITIALIZE SENT TO IMPOSSIBLE
3645 00004ECF A2[05770100]     <1>     mov     [SEEK_STATUS], al       ; INDICATE RECALIBRATE NEEDED
3646 00004ED4 A2[07770100]     <1>     mov     [MOTOR_COUNT], al       ; INITIALIZE MOTOR COUNT
3647 00004ED9 A2[06770100]     <1>     mov     [MOTOR_STATUS], al      ; INITIALIZE DRIVES TO OFF STATE
3648 00004EDE A2[08770100]     <1>     mov     [DSKETTE_STATUS], al    ; NO ERRORS
3649                            <1>     ;
3650                            <1>     ; 28/02/2015
3651                            <1>     ;mov    word [cfd], 100h
3652 00004EE3 E815F4FFFF       <1>     call    DSK_RESET
3653 00004EE8 5A               <1>     pop     edx
3654 00004EE9 F8               <1>     clc     ; 29/05/2016
3655 00004EEA C3               <1>     retn
3656                            <1>
3657                            <1> ;SUP0:
3658                            <1> ;   call    DRIVE_DET               ; DETERMINE DRIVE
3659                            <1> ;   call    XLAT_OLD                ; TRANSLATE STATE TO COMPATIBLE MODE
3660                            <1> ;   ; 02/01/2015
3661                            <1> ;   ;inc    di                      ; POINT TO NEXT DRIVE
3662                            <1> ;   ;cmp    di, MAX_DRV             ; SEE IF DONE
3663                            <1> ;   ;jnz    short SUP0              ; REPEAT FOR EACH ORIVE
3664                            <1> ;       cmp     byte [fd1_type], 0
3665                            <1> ;   jna     short sup1
3666                            <1> ;   or      di, di
3667                            <1> ;   jnz     short sup1
3668                            <1> ;   inc     di
3669                            <1> ;       jmp     short SUP0
3670                            <1> ;sup1:
3671                            <1> ;   mov     byte [SEEK_STATUS], 0 ; FORCE RECALIBRATE
3672                            <1> ;   ;and    byte [RTC_WAIT_FLAG], 0FEh ; ALLOW FOR RTC WAIT
3673                            <1> ;   call    SETUP_END               ; VARIOUS CLEANUPS
3674                            <1> ;   ;;pop   ds                      ; RESTORE CALLERS REGISTERS
3675                            <1> ;   ;pop    di
3676                            <1> ;   pop     edx
3677                            <1> ;   ;pop    cx
3678                            <1> ;   ;pop    bx
3679                            <1> ;   ;pop    ax
3680                            <1> ;   retn
3681                            <1>
3682                            <1> ;////////////////////////////////////////////////////
3683                            <1> ;; END OF DISKETTE I/O ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
3684                            <1>
3685                            <1> ; 17/04/2021 (TRDOS 386 v2.0.4)
3686                            <1>
3687                            <1> ; 11/04/2021
3688                            <1> ;int13h: ; 21/02/2015
3689                            <1>     ;pushfd
3690                            <1>     ;push   cs
3691                            <1>     ;;call  DISK_IO
3692                            <1>     ;;retn
3693                            <1>
3694                            <1> ;;;;;; DISK I/O ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; 21/02/2015 ;;;
3695                            <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> SYS1equ 61h        ; PORT_B  (diskette.inc)
                        <1>
                        <1> ; 23/12/2014
                        <1> %define CMD_BLOCK       ebp-8  ; 21/02/2015
                        <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                                                            :
                        <1> ;   (AH)= 0FH  UNUSED                                                            :
                        <1> ;   (AH)= 10H  TEST DRIVE READY                                                  :
                        <1> ;   (AH)= 11H  RECALIBRATE                                                       :
                        <1> ;   (AH)= 12H  UNUSED                                                            :
                        <1> ;   (AH)= 13H  UNUSED                                                            :
                        <1> ;   (AH)= 14H  CONTROLLER INTERNAL DIAGNOSTIC                                    :
                        <1> ;   (AH)= 15H  READ DASD TYPE                                                    :
                        <1> ;                                                                                :
                        <1> ;--------------------------------------------------------------------------------
                        <1> ;                                                                                :
                        <1> ;   REGISTERS USED FOR FIXED DISK OPERATIONS                                     :
                        <1> ;                                                                                :
                        <1> ;          (DL)    -  DRIVE NUMBER     (80H-81H FOR DISK. VALUE CHECKED)  :
                        <1> ;          (DH)    -  HEAD NUMBER     (0-15 ALLOWED, NOT VALUE CHECKED)  :
                        <1> ;          (CH)    -  CYLINDER NUMBER  (0-1023, NOT VALUE CHECKED)(SEE CL):
                        <1> ;          (CL)    -  SECTOR NUMBER    (1-17, NOT VALUE CHECKED)              :
                        <1> ;                                                                                :
                        <1> ;                     NOTE: HIGH 2 BITS OF CYLINDER NUMBER ARE PLACED     :
                        <1> ;                          IN THE HIGH 2 BITS OF THE CL REGISTER             :
                        <1> ;                          (10 BITS TOTAL)                                       :
                        <1> ;                                                                                :
                        <1> ;          (AL)    -  NUMBER OF SECTORS (MAXIMUM POSSIBLE RANGE 1-80H,    :
                        <1> ;                                        FOR READ/WRITE LONG 1-79H)      :
                        <1> ;                                                                                :
                        <1> ;          (ES:BX) -  ADDRESS OF BUFFER FOR READS AND WRITES,                    :
                        <1> ;                     (NOT REQUIRED FOR VERIFY)                                  :
                        <1> ;                                                                                :
                        <1> ;          FORMAT (AH=5) ES:BX POINTS TO A 512 BYTE BUFFER. THE FIRST     :
                        <1> ;                     2*(SECTORS/TRACK) BYTES CONTAIN F,N FOR EACH SECTOR.:
                        <1> ;                     F = 00H FOR A GOOD SECTOR                                  :
                        <1> ;                         80H FOR A BAD SECTOR                                   :
                        <1> ;                     N = SECTOR NUMBER                                          :
                        <1> ;                     FOR AN INTERLEAVE OF 2 AND 17 SECTORS/TRACK            :
                        <1> ;                     THE TABLE SHOULD BE:                                       :
                        <1> ;                                                                                :
                        <1> ;             DB      00H,01H,00H,0AH,00H,02H,00H,0BH,00H,03H,00H,0CH     :
                        <1> ;             DB      00H,04H,00H,0DH,00H,05H,00H,0EH,00H,06H,00H,0FH     :
                        <1> ;             DB      00H,07H,00H,10H,00H,08H,00H,11H,00H,09H                    :
                        <1> ;                                                                                :
                        <1> ;--------------------------------------------------------------------------------
                        <1>
                        <1> ;--------------------------------------------------------------------------------
                        <1> ; OUTPUT                                                                         :
                        <1> ;   AH = STATUS OF CURRENT OPERATION                                             :
                        <1> ;        STATUS BITS ARE DEFINED IN THE EQUATES BELOW                            :
                        <1> ;   CY = 0  SUCCESSFUL OPERATION (AH=0 ON RETURN)                                :
                        <1> ;   CY = 1  FAILED OPERATION (AH HAS ERROR REASON)                               :
```

```
                        <1> ;                                                                      :
                        <1> ;   NOTE:   ERROR 11H  INDICATES THAT THE DATA READ HAD A RECOVERABLE      :
                        <1> ;           ERROR WHICH WAS CORRECTED BY THE ECC ALGORITHM.  THE DATA       :
                        <1> ;           IS PROBABLY GOOD,   HOWEVER THE BIOS ROUTINE INDICATES AN      :
                        <1> ;           ERROR TO ALLOW THE CONTROLLING PROGRAM A CHANCE TO DECIDE       :
                        <1> ;           FOR ITSELF.  THE  ERROR  MAY  NOT  RECUR  IF  THE DATA IS      :
                        <1> ;           REWRITTEN.                                                     :
                        <1> ;                                                                          :
                        <1> ;   IF DRIVE PARAMETERS WERE REQUESTED (DL >= 80H),                        :
                        <1> ;      INPUT:                                                              :
                        <1> ;        (DL) = DRIVE NUMBER                                               :
                        <1> ;        ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)                                                 :
                        <1> ;        EBX = Buffer address for fixed disk parameters table (32 bytes)   :
                        <1> ;      OUTPUT:                                                             :
                        <1> ;        (DL) = NUMBER OF CONSECUTIVE ACKNOWLEDGING DRIVES ATTACHED (1-2)  :
                        <1> ;               (CONTROLLER CARD ZERO TALLY ONLY)                            :
                        <1> ;        (DH) = MAXIMUM USEABLE VALUE FOR HEAD NUMBER                      :
                        <1> ;        (CH) = MAXIMUM USEABLE VALUE FOR CYLINDER NUMBER                        :
                        <1> ;        (CL) = MAXIMUM USEABLE VALUE FOR SECTOR NUMBER                    :
                        <1> ;              AND CYLINDER NUMBER HIGH BITS                               :
                        <1> ;                                                                          :
                        <1> ;   IF READ DASD TYPE WAS REQUESTED,                                       :
                        <1> ;                                                                          :
                        <1> ;   AH = 0 - NOT PRESENT                                                   :
                        <1> ;        1 - DISKETTE - NO CHANGE LINE AVAILABLE                           :
                        <1> ;        2 - DISKETTE - CHANGE LINE AVAILABLE                              :
                        <1> ;        3 - FIXED DISK                                                    :
                        <1> ;                                                                          :
                        <1> ;   CX,DX = NUMBER OF 512 BYTE BLOCKS WHEN AH = 3                              :
                        <1> ;                                                                          :
                        <1> ;   REGISTERS WILL BE PRESERVED EXCEPT WHEN THEY ARE USED TO RETURN         :
                        <1> ;   INFORMATION.                                                           :
                        <1> ;                                                                          :
                        <1> ;   NOTE: IF AN ERROR IS REPORTED BY THE DISK CODE, THE APPROPRIATE         :
                        <1> ;          ACTION IS TO RESET THE DISK, THEN RETRY THE OPERATION.             :
                        <1> ;                                                                          :
                        <1> ;--------------------------------------------------------------------------
                        <1>
                        <1> SENSE_FAIL  EQU     0FFH            ; NOT IMPLEMENTED
                        <1> NO_ERR              EQU     0E0H            ; STATUS ERROR/ERROR REGISTER=0
                        <1> WRITE_FAULT EQU     0CCH            ; WRITE FAULT ON SELECTED DRIVE
                        <1> UNDEF_ERR   EQU     0BBH            ; UNDEFINED ERROR OCCURRED
                        <1> NOT_RDY     EQU     0AAH            ; DRIVE NOT READY
                        <1> TIME_OUT    EQU     80H             ; ATTACHMENT FAILED TO RESPOND
                        <1> BAD_SEEK    EQU     40H             ; SEEK OPERATION FAILED
                        <1> BAD_CNTLR   EQU     20H             ; CONTROLLER HAS FAILED
                        <1> DATA_CORRECTED      EQU     11H             ; ECC CORRECTED DATA ERROR
                        <1> BAD_ECC     EQU     10H             ; BAD ECC ON DISK READ
                        <1> BAD_TRACK   EQU     0BH             ; NOT IMPLEMENTED
                        <1> BAD_SECTOR  EQU     0AH             ; BAD SECTOR FLAG DETECTED
                        <1> ;DMA_BOUNDARY       EQU     09H             ; DATA EXTENDS TOO FAR
                        <1> INIT_FAIL   EQU     07H             ; DRIVE PARAMETER ACTIVITY FAILED
                        <1> BAD_RESET   EQU     05H             ; RESET FAILED
                        <1> ;RECORD_NOT_FND     EQU     04H             ; REQUESTED SECTOR NOT FOUND
                        <1> ;BAD_ADDR_MARK      EQU     02H             ; ADDRESS MARK NOT FOUND
                        <1> ;BAD_CMD    EQU     01H             ; BAD COMMAND PASSED TO DISK I/O
                        <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>
00004EEB 90             <1> align 2
                        <1>
                        <1> ;-----              STATUS REGISTER
                        <1>
                        <1> ST_ERROR    EQU     00000001B       ;
```

```
3944                                    <1> ST_INDEX    EQU     00000010B       ;
3945                                    <1> ST_CORRCTD  EQU     00000100B       ; ECC CORRECTION SUCCESSFUL
3946                                    <1> ST_DRQ              EQU     00001000B       ;
3947                                    <1> ST_SEEK_COMPL       EQU     00010000B       ; SEEK COMPLETE
3948                                    <1> ST_WRT_FLT  EQU     00100000B       ; WRITE FAULT
3949                                    <1> ST_READY    EQU     01000000B       ;
3950                                    <1> ST_BUSY     EQU     10000000B       ;
3951                                    <1>
3952                                    <1> ;-----              ERROR REGISTER
3953                                    <1>
3954                                    <1> ERR_DAM     EQU     00000001B       ; DATA ADDRESS MARK NOT FOUND
3955                                    <1> ERR_TRK_0   EQU     00000010B       ; TRACK 0 NOT FOUND ON RECAL
3956                                    <1> ERR_ABORT   EQU     00000100B       ; ABORTED COMMAND
3957                                    <1> ;           EQU     00001000B       ; NOT USED
3958                                    <1> ERR_ID              EQU     00010000B       ; ID NOT FOUND
3959                                    <1> ;           EQU     00100000B       ; NOT USED
3960                                    <1> ERR_DATA_ECC        EQU     01000000B
3961                                    <1> ERR_BAD_BLOCK       EQU     10000000B
3962                                    <1>
3963                                    <1>
3964                                    <1> RECAL_CMD   EQU     00010000B       ; DRIVE RECAL  (10H)
3965                                    <1> READ_CMD    EQU     00100000B       ;       READ   (20H)
3966                                    <1> WRITE_CMD   EQU     00110000B       ;       WRITE  (30H)
3967                                    <1> VERIFY_CMD  EQU     01000000B       ;       VERIFY (40H)
3968                                    <1> FMTTRK_CMD  EQU     01010000B       ; FORMAT TRACK (50H)
3969                                    <1> INIT_CMD    EQU     01100000B       ;   INITIALIZE (60H)
3970                                    <1> SEEK_CMD    EQU     01110000B       ;       SEEK   (70H)
3971                                    <1> DIAG_CMD    EQU     10010000B       ; DIAGNOSTIC   (90H)
3972                                    <1> SET_PARM_CMD        EQU     10010001B       ; DRIVE PARMS  (91H)
3973                                    <1> NO_RETRIES  EQU     00000001B       ; CHD MODIFIER (01H)
3974                                    <1> ECC_MODE    EQU     00000010B       ; CMD MODIFIER (02H)
3975                                    <1> BUFFER_MODE EQU     00001000B       ; CMD MODIFIER (08H)
3976                                    <1>
3977                                    <1> ;MAX_FILE   EQU     2
3978                                    <1> ;S_MAX_FILE EQU     2
3979                                    <1> MAX_FILE    equ     4               ; 22/12/2014
3980                                    <1> S_MAX_FILE  equ     4               ; 22/12/2014
3981                                    <1>
3982                                    <1> DELAY_1     EQU     25H             ; DELAY FOR OPERATION COMPLETE
3983                                    <1> DELAY_2     EQU     0600H           ; DELAY FOR READY
3984                                    <1> DELAY_3     EQU     0100H           ; DELAY FOR DATA REQUEST
3985                                    <1>
3986                                    <1> HF_FAIL     EQU     08H             ; CMOS FLAG IN BYTE 0EH
3987                                    <1>
3988                                    <1> ;-----              COMMAND BLOCK REFERENCE
3989                                    <1>
3990                                    <1> ;CMD_BLOCK      EQU     BP-8            ; @CMD_BLOCK REFERENCES BLOCK HEAD IN SS
3991                                    <1>                                     ;  (BP) POINTS TO COMMAND BLOCK TAIL
3992                                    <1>                                     ;       AS DEFINED BY THE "ENTER" PARMS
3993                                    <1> ; 19/12/2014
3994                                    <1> ORG_VECTOR  equ     4*13h           ; INT 13h vector
3995                                    <1> DISK_VECTOR equ     4*40h           ; INT 40h vector (for floppy disks)
3996                                    <1> ;HDISK_INT  equ     4*76h           ; Primary HDC - Hardware interrupt (IRQ14)
3997                                    <1> ;HDISK_INT1 equ     4*76h           ; Primary HDC - Hardware interrupt (IRQ14)
3998                                    <1> ;HDISK_INT2 equ     4*77h           ; Secondary HDC - Hardware interrupt (IRQ15)
3999                                    <1> ;HF_TBL_VEC equ     4*41h           ; Pointer to 1st fixed disk parameter table
4000                                    <1> ;HF1_TBL_VEC        equ     4*46h           ; Pointer to 2nd fixed disk parameter table
4001                                    <1>
4002                                    <1> align 2
4003                                    <1>
4004                                    <1> ;--------------------------------------------------------------
4005                                    <1> ; FIXED DISK I/O SETUP                                          :
4006                                    <1> ;                                                       :
4007                                    <1> ;  -  ESTABLISH TRANSFER VECTORS FOR THE FIXED DISK             :
4008                                    <1> ;  -  PERFORM POWER ON DIAGNOSTICS                              :
4009                                    <1> ;     SHOULD AN ERROR OCCUR A "1701" MESSAGE IS DISPLAYED       :
4010                                    <1> ;                                                               :
4011                                    <1> ;--------------------------------------------------------------
4012                                    <1>
4013                                    <1> ; 09/08/2022
4014                                    <1> ; 06/08/2022 - TRDOS 386 Kernel v2.0.5
4015                                    <1> ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
4016                                    <1>
4017                                    <1> DISK_SETUP:
4018                                    <1>     ;CLI
4019                                    <1>     ;;mov   ax, ABS0                ; GET ABSOLUTE SEGMENT
4020                                    <1>     ;xor    ax, ax
4021                                    <1>     ;mov    ds, ax                  ; SET SEGMENT REGISTER
4022                                    <1>     ;mov    ax, [ORG_VECTOR]        ; GET DISKETTE VECTOR
4023                                    <1>     ;mov    [DISK_VECTOR], ax       ;  INTO INT 40H
4024                                    <1>     ;mov    ax, [ORG_VECTOR+2]
4025                                    <1>     ;mov    [DISK_VECTOR+2], ax
4026                                    <1>     ;mov    word [ORG_VECTOR], DISK_IO ; FIXED DISK HANDLER
4027                                    <1>     ;mov    [ORG_VECTOR+2], cs
4028                                    <1>     ; 1st controller (primary master, slave)   - IRQ 14
4029                                    <1>     ;;mov   word [HDISK_INT], HD_INT    ; FIXED DISK INTERRUPT
4030                                    <1>     ;mov    word [HDISK_INT1], HD_INT   ;
4031                                    <1>     ;;mov   [HDISK_INT+2], cs
4032                                    <1>     ;mov    [HDISK_INT1+2], cs
4033                                    <1>     ; 2nd controller (secondary master, slave) - IRQ 15
4034                                    <1>     ;mov    word [HDISK_INT2], HD1_INT ;
4035                                    <1>     ;mov    [HDISK_INT2+2], cs
4036                                    <1>     ;
4037                                    <1>     ;;mov   word [HF_TBL_VEC], HD0_DPT    ; PARM TABLE DRIVE 80
4038                                    <1>     ;;mov   word [HF_TBL_VEC+2], DPT_SEGM
4039                                    <1>     ;;mov   word [HF1_TBL_VEC], HD1_DPT   ; PARM TABLE DRIVE 81
4040                                    <1>     ;;mov   word [HF1_TBL_VEC+2], DPT_SEGM
4041                                    <1>     ;push   cs
4042                                    <1>     ;pop    ds
4043                                    <1>     ;mov    word [HDPM_TBL_VEC], HD0_DPT  ; PARM TABLE DRIVE 80h
4044                                    <1>     ;mov    word [HDPM_TBL_VEC+2], DPT_SEGM
4045 00004EEC C705[1C770100]0000-       <1>     mov     dword [HDPM_TBL_VEC], (DPT_SEGM*16)+HD0_DPT
4045 00004EF4 0900
4046                                    <1>     ;mov    word [HDPS_TBL_VEC], HD1_DPT  ; PARM TABLE DRIVE 81h
4047                                    <1>     ;mov    word [HDPS_TBL_VEC+2], DPT_SEGM
4048 00004EF6 C705[20770100]2000-       <1>     mov     dword [HDPS_TBL_VEC], (DPT_SEGM*16)+HD1_DPT
4048 00004EFE 0900
4049                                    <1>     ;mov    word [HDSM_TBL_VEC], HD2_DPT  ; PARM TABLE DRIVE 82h
4050                                    <1>     ;mov    word [HDSM_TBL_VEC+2], DPT_SEGM
4051 00004F00 C705[24770100]4000-       <1>     mov     dword [HDSM_TBL_VEC], (DPT_SEGM*16)+HD2_DPT
4051 00004F08 0900
4052                                    <1>     ;mov    word [HDSS_TBL_VEC], HD3_DPT  ; PARM TABLE DRIVE 83h
4053                                    <1>     ;mov    word [HDSS_TBL_VEC+2], DPT_SEGM
4054 00004F0A C705[28770100]6000-       <1>     mov     dword [HDSS_TBL_VEC], (DPT_SEGM*16)+HD3_DPT
4054 00004F12 0900
4055                                    <1>     ;
4056                                    <1>     ;;in    al, INTB01              ; TURN ON SECOND INTERRUPT CHIP
4057                                    <1>     ;;;and  al, 0BFh
4058                                    <1>     ;;and   al, 3Fh                 ; enable IRQ 14 and IRQ 15
4059                                    <1>     ;;;jmp  $+2
4060                                    <1>     ;;IODELAY
4061                                    <1>     ;;out   INTB01, al
4062                                    <1>     ;;IODELAY
4063                                    <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
00004F14 BE[7E660000]   <1>     mov     esi, hd0_type
                        <1>     ;;mov   cx, 4
                        <1>     ;mov    ecx, 4
                        <1>     ; 06/08/2022
00004F19 29C9           <1>     sub     ecx, ecx
00004F1B B104           <1>     mov     cl, 4
                        <1> hde_l:
00004F1D AC             <1>     lodsb
00004F1E 3C80           <1>     cmp     al, 80h                 ; 8?h = existing
00004F20 7206           <1>     jb      short _L4
00004F22 FE05[18770100] <1>     inc     byte [HF_NUM]           ; + 1 hard (fixed) disk drives
                        <1> _L4: ; 26/02/2015
00004F28 E2F3           <1>     loop    hde_l
                        <1> ;_L4:                                       ; 0 <= [HF_NUM] =< 4
                        <1> ;L4:
                        <1>     ;; 31/12/2014 - cancel controller diagnostics here
                        <1>     ;;;mov  cx, 3  ; 26/12/2014 (Award BIOS 1999)
                        <1>     ;;mov   cl, 3
                        <1>     ;;
                        <1>     ;;mov   DL, 80H                 ; CHECK THE CONTROLLER
                        <1> ;;hdc_dl:
                        <1>     ;;mov   AH, 14H                 ; USE CONTROLLER DIAGNOSTIC COMMAND
                        <1>     ;;INT   13H                     ; CALL BIOS WITH DIAGNOSTIC COMMAND
                        <1>     ;;;jc   short CTL_ERRX          ; DISPLAY ERROR MESSAGE IF BAD RETURN
                        <1>     ;;;jc   short POD_DONE ;22/12/2014
                        <1>     ;;jnc   short hdc_reset0
                        <1>     ;;loop  hdc_dl
                        <1>     ;;; 27/12/2014
                        <1>     ;;stc
                        <1>     ;;retn
                        <1>     ;
                        <1> ;;hdc_reset0:
                        <1>     ; 18/01/2015
00004F2A 8A0D[18770100] <1>     mov     cl, [HF_NUM]
00004F30 20C9           <1>     and     cl, cl
00004F32 740D           <1>     jz      short POD_DONE
                        <1>     ;
00004F34 B27F           <1>     mov     dl, 7Fh
                        <1> hdc_reset1:
00004F36 FEC2           <1>     inc     dl
                        <1>     ;; 31/12/2015
                        <1>     ;;push  dx
                        <1>     ;;push  cx
                        <1>     ;;push  ds
                        <1>     ;;sub   ax, ax
                        <1>     ;;mov   ds, ax
                        <1>     ;;mov   ax, [TIMER_LOW]              ; GET START TIMER COUNTS
                        <1>     ;;pop   ds
                        <1>     ;;mov   bx, ax
                        <1>     ;;add   ax, 6*182               ; 60 SECONDS* 18.2
                        <1>     ;;mov   cx, ax
                        <1>     ;;mov   word [wait_count], 0  ; 22/12/2014 (reset wait counter)
                        <1>     ;;
                        <1>     ;; 31/12/2014 - cancel HD_RESET_1
                        <1>     ;;call  HD_RESET_1              ; SET UP DRIVE 0, (1,2,3)
                        <1>     ;;pop   cx
                        <1>     ;;pop   dx
                        <1>     ;;
                        <1>     ; 18/01/2015
00004F38 B40D           <1>     mov     ah, 0Dh ; ALTERNATE RESET
                        <1>     ;int    13h
00004F3A E8CC000000     <1>     call    int13h
00004F3F E2F5           <1>     loop    hdc_reset1
                        <1>     ;clc    ; 29/05/2016
                        <1> POD_DONE:
00004F41 C3             <1>     retn
                        <1>
                        <1> ;;-----     POD_ERROR
                        <1>
                        <1> ;;CTL_ERRX:
                        <1> ;   ;mov    SI,OFFSET F1782         ; CONTROLLER ERROR
                        <1> ;   ;call   SET_FAIL                ; DO NOT IPL FROM DISK
                        <1> ;   ;call   E_MSG                   ; DISPLAY ERROR AND SET (BP) ERROR FLAG
                        <1> ;   ;jmp    short POD_DONE
                        <1>
                        <1> ;;HD_RESET_1:
                        <1> ;;  ;push   BX                      ; SAVE TIMER LIMITS
                        <1> ;;  ;push   CX
                        <1> ;;RES_1: mov       AH,09H                ; SET DRIVE PARAMETERS
                        <1> ;;  INT     13H
                        <1> ;;  jc      short RES_2
                        <1> ;;  mov     AH,11h                  ; RECALIBRATE DRIVE
                        <1> ;;  INT     13H
                        <1> ;;  jnc     short RES_CK            ; DRIVE OK
                        <1> ;;RES_2: ;call     POD_TCHK              ; CHECK TIME OUT
                        <1> ;;  cmp     word [wait_count], 6*182 ; waiting time (in timer ticks)
                        <1> ;;                                  ; (30 seconds)
                        <1> ;;  ;cmc
                        <1> ;;  ;jnc    short RES_1
                        <1> ;;  jb      short RES_1
                        <1> ;;;RES_FL: ;mov    SI,OFFSET F1781         ; INDICATE DISK 1 FAILURE;
                        <1> ;;  ;test   DL,1
                        <1> ;;  ;jnz    short RES_E1
                        <1> ;;  ;mov    SI,OFFSET F1780         ; INDICATE DISK 0 FAILURE
                        <1> ;;  ;call   SET_FAIL                ; DO NOT TRY TO IPL DISK 0
                        <1> ;;  ;jmp    SHORT RES_E1
                        <1> ;;RES_ER: ; 22/12/2014
                        <1> ;;RES_OK:
                        <1> ;;  ;pop    CX                      ; RESTORE TIMER LIMITS
                        <1> ;;  ;pop    BX
                        <1> ;;  retn
                        <1> ;;
                        <1> ;;RES_RS: mov      AH,00H                ; RESET THE DRIVE
                        <1> ;;  INT     13H
                        <1> ;;RES_CK: mov      AH,08H                ; GET MAX CYLINDER,HEAD,SECTOR
                        <1> ;;  mov     BL,DL                   ; SAVE DRIVE CODE
                        <1> ;;  INT     13H
                        <1> ;;  jc      short RES_ER
```

```
                        <1> ;;  mov     [NEC_STATUS],CX       ; SAVE MAX CYLINDER, SECTOR
                        <1> ;;  mov     DL,BL                 ; RESTORE DRIVE CODE
                        <1> ;;RES_3: mov        AX,0401H              ; VERIFY THE LAST SECTOR
                        <1> ;;  INT     13H
                        <1> ;;  jnc     short RES_OK          ; VERIFY OK
                        <1> ;;  cmp     AH,BAD_SECTOR         ; OK ALSO IF JUST ID READ
                        <1> ;;  JE      short RES_OK
                        <1> ;;  cmp     AH,DATA_CORRECTED
                        <1> ;;  JE      short RES_OK
                        <1> ;;  cmp     AH,BAD_ECC
                        <1> ;;  JE      short RES_OK
                        <1> ;;  ;call   POD_TCHK              ; CHECK FOR TIME OUT
                        <1> ;;  cmp     word [wait_count], 6*182 ; waiting time (in timer ticks)
                        <1> ;;                                ; (60 seconds)
                        <1> ;;  cmc
                        <1> ;;  jc      short RES_ER          ; FAILED
                        <1> ;;  mov     CX,[NEC_STATUS]       ; GET SECTOR ADDRESS, AND CYLINDER
                        <1> ;;  mov     AL,CL                 ; SEPARATE OUT SECTOR NUMBER
                        <1> ;;  and     AL,3FH
                        <1> ;;  dec     AL                    ; TRY PREVIOUS ONE
                        <1> ;;  jz      short RES_RS          ; WE'VE TRIED ALL SECTORS ON TRACK
                        <1> ;;  and     CL,0C0H               ; KEEP CYLINDER BITS
                        <1> ;;  OR      CL,AL                 ; MERGE SECTOR WITH CYLINDER BITS
                        <1> ;;  mov     [NEC_STATUS],CX       ; SAVE CYLINDER, NEW SECTOR NUMBER
                        <1> ;;  jmp     short RES_3           ; TRY AGAIN
                        <1> ;;;RES_ER: mov      SI,OFFSET F1791       ; INDICATE DISK 1 ERROR
                        <1> ;;  ;test   DL,1
                        <1> ;;  ;jnz    short RES_E1
                        <1> ;;  ;mov    SI,OFFSET F1790       ; INDICATE DISK 0 ERROR
                        <1> ;;;RES_E1:
                        <1> ;;  ;call   E_MSG                 ; DISPLAY ERROR AND SET (BP) ERROR FLAG
                        <1> ;;;RES_OK:
                        <1> ;;  ;pop    CX                    ; RESTORE TIMER LIMITS
                        <1> ;;  ;pop    BX
                        <1> ;;  ;retn
                        <1> ;
                        <1> ;;SET_FAIL:
                        <1> ;   ;mov    AX,X*(CMOS_DIAG+NMI)  ; GET CMOS ERROR BYTE
                        <1> ;   ;call   CMOS_READ
                        <1> ;   ;OR     AL,HF_FAIL            ; SET DO NOT IPL FROM DISK FLAG
                        <1> ;   ;xchg   AH,AL                 ; SAVE IT
                        <1> ;   ;call   CMOS_WRITE            ; PUT IT OUT
                        <1> ;   ;retn
                        <1> ;
                        <1> ;;POD_TCHK:                       ; CHECK FOR 30 SECOND TIME OUT
                        <1> ;   ;pop    AX                    ; SAVE RETURN
                        <1> ;   ;pop    CX                    ; GET TIME OUT LIMITS
                        <1> ;   ;pop    BX
                        <1> ;   ;push   BX                    ; AND SAVE THEM AGAIN
                        <1> ;   ;push   CX
                        <1> ;   ;push   AX
                        <1> ;   ;push   ds
                        <1> ;   ;xor    ax, ax
                        <1> ;   ;mov    ds, ax                ; RESTORE RETURN
                        <1> ;   ;mov    AX, [TIMER_LOW]               ; AX = CURRENT TIME
                        <1> ;   ;                             ; BX = START TIME
                        <1> ;   ;                             ; CX = END TIME
                        <1> ;   ;pop    ds
                        <1> ;   ;cmp    BX,CX
                        <1> ;   ;JB     short TCHK1           ; START < END
                        <1> ;   ;cmp    BX,AX
                        <1> ;   ;JB     short TCHKG           ; END < START < CURRENT
                        <1> ;   ;jmp    SHORT TCHK2           ; END, CURRENT < START
                        <1> ;;TCHK1: cmp        AX,BX
                        <1> ;;  JB      short TCHKNG          ; CURRENT < START < END
                        <1> ;;TCHK2: cmp        AX,CX
                        <1> ;;  JB      short TCHKG           ; START < CURRENT < END
                        <1> ;;                                ; OR CURRENT < END < START
                        <1> ;;TCHKNG: STC                             ; CARRY SET INDICATES TIME OUT
                        <1> ;;  retn
                        <1> ;;TCHKG: CLC                              ; INDICATE STILL TIME
                        <1> ;;  retn
                        <1> ;;
                        <1> ;;int_13h:
                        <1>
                        <1> ;---------------------------------------
                        <1> ;   FIXED DISK BIOS ENTRY POINT   :
                        <1> ;---------------------------------------
                        <1>
                        <1> ; 17/07/2022
                        <1> ; 16/07/2022
                        <1> ; 13/07/2022 - TRDOS 386 v2.0.5
                        <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
00004F42 80642408FE     <1>     and     byte [esp+8], 11111110b  ; clear carry bit of eflags register
                        <1>
                        <1>     ; 13/07/2022
00004F47 06             <1>     push    es
                        <1>     ; 16/05/2016
00004F48 1E             <1>     push    ds
00004F49 53             <1>     push    ebx ; user's buffer address (virtual)
                        <1>     ; 13/07/2022
00004F4A 51             <1>     push    ecx
00004F4B 52             <1>     push    edx
00004F4C 56             <1>     push    esi
00004F4D 57             <1>     push    edi
                        <1>
                        <1>     ;mov    bx, KDATA ; System (Kernel's) data segment
                        <1>     ;mov    ds, bx
                        <1>     ; 13/07/2022
00004F4E 66BF1000       <1>     mov     di, KDATA
00004F52 8EDF           <1>     mov     ds, di
00004F54 8EC7           <1>     mov     es, di
                        <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>
                        <1>     ; 13/07/2022
                        <1>     ;pop    dword [user_buffer] ; 01/06/2016
00004F56 891D[0C830100] <1>     mov     [user_buffer], ebx
                        <1>
00004F5C C605[467C0100]00 <1>   mov     byte [scount], 0 ; sector count for transfer
00004F63 80FC03         <1>     cmp     ah, 03h ; chs write
00004F66 773C           <1>     ja      short int33h_2
```

```
4312 00004F68 7407              <1>     je      short int33h_0
4313 00004F6A 80FC02            <1>     cmp     ah, 02h ; chs read
4314 00004F6D 7267              <1>     jb      short int33h_5
4315 00004F6F EB5A              <1>     jmp     short int33h_4
4316                            <1> int33h_0:
4317                            <1>     ;; 17/07/2022 - 64K r/w buffer limit check ?
4318                            <1>     ;cmp    al, 80h ; 128
4319                            <1>     ;ja     short int33h_8 ; error
4320                            <1>     ;; 17/07/2022 - zero r/w count check ?
4321                            <1>     ;or     al, al
4322                            <1>     ;jz     short int33h_8 ; error
4323                            <1>
4324                            <1>     ; 17/07/2022 (buffer limit and zero count check)
4325 00004F71 FEC8              <1>     dec     al
4326 00004F73 781A              <1>     js      short int33h_8 ; error
4327 00004F75 FEC0              <1>     inc     al
4328                            <1>
4329                            <1>     ; transfer user's buffer content to sector buffer
4330 00004F77 51                <1>     push    ecx
4331 00004F78 0FB6C8            <1>     movzx   ecx, al
4332                            <1> int33h_1:
4333                            <1>     ; 13/07/2022
4334                            <1>     ;push   esi
4335                            <1>     ;mov    esi, [user_buffer]
4336 00004F7B 89DE              <1>     mov     esi, ebx
4337                            <1>     ; esi = user's buffer address (virtual, ebx)
4338                            <1>     ;push   edi
4339                            <1>     ;push   es
4340 00004F7D 50                <1>     push    eax
4341                            <1>     ;mov    ax, KDATA
4342                            <1>     ;mov    es, ax
4343 00004F7E BF00000700        <1>     mov     edi, Cluster_Buffer
4344 00004F83 C1E109            <1>     shl     ecx, 9 ; * 512
4345 00004F86 E8A9BB0000        <1>     call    transfer_from_user_buffer
4346                            <1>             ; (ecx and eax will be modified)
4347 00004F8B 58                <1>     pop     eax
4348                            <1>     ; 13/07/2022
4349                            <1>     ;pop    es
4350                            <1>     ;pop    edi
4351                            <1>     ;pop    esi
4352 00004F8C 59                <1>     pop     ecx
4353 00004F8D 7347              <1>     jnc     short int33h_5
4354                            <1>
4355                            <1>     ;mov    ebx, [user_buffer] ; 01/06/2016
4356                            <1>     ;pop    ds
4357                            <1>
4358                            <1> int33h_8:
4359                            <1>     ; 13/07/2022
4360 00004F8F B8FF000000        <1>     mov     eax, 0FFh ; Unknown error !?
4361                            <1> int33h_9:
4362 00004F94 5F                <1>     pop     edi
4363 00004F95 5E                <1>     pop     esi
4364 00004F96 5A                <1>     pop     edx
4365 00004F97 59                <1>     pop     ecx
4366 00004F98 5B                <1>     pop     ebx
4367 00004F99 1F                <1>     pop     ds
4368 00004F9A 07                <1>     pop     es
4369                            <1>
4370                            <1>     ; 13/07/2022
4371 00004F9B 7305              <1>     jnc     short int33h_7
4372                            <1>
4373                            <1>     ;;15/01/2017
4374                            <1>     ; 02/01/2017
4375                            <1>     ;cli
4376                            <1>     ;;mov   byte [ss:intflg], 0 ; 07/01/2017
4377                            <1>     ;
4378                            <1>     ; (*) 29/05/2016
4379                            <1>     ; (*) retf 4 ; skip eflags on stack
4380                            <1>
4381                            <1>     ; 29/05/2016 -set carry flag on stack-
4382                            <1>     ; [esp] = EIP
4383                            <1>     ; [esp+4] = CS
4384                            <1>     ; [esp+8] = E-FLAGS
4385 00004F9D 804C240801          <1>     or      byte [esp+8], 1  ; set carry bit of eflags register
4386                            <1>     ; [esp+12] = ESP (user)
4387                            <1>     ; [esp+16] = SS (User)
4388                            <1>     ;
4389                            <1>     ; 13/07/2022
4390                            <1> int33h_7:
4391 00004FA2 FA                <1>     cli
4392                            <1>     ;;15/01/2017
4393                            <1>     ;;mov   byte [ss:intflg], 0 ; 07/01/2017
4394                            <1>     ; cf = 0  ; use eflags which is in stack
4395 00004FA3 CF                <1>     iretd
4396                            <1>
4397                            <1>     ; (*) 29/05/2016 - 'retf 4' intruction causes to stack fault
4398                            <1>     ; (OUTER-PRIVILEGE-LEVEL)
4399                            <1>     ; INTEL 80386 PROGRAMMER'S REFERENCE MANUAL 1986
4400                            <1>     ; // RETF instruction:
4401                            <1>     ;
4402                            <1>     ; IF OperandMode=32 THEN
4403                            <1>     ;    Load CS:EIP from stack;
4404                            <1>     ;    Set CS RPL to CPL;
4405                            <1>     ;    Increment ESP by 8 plus the immediate offset if it exists;
4406                            <1>     ;    Load SS:ESP from stack;
4407                            <1>     ; ELSE (* OperandMode=16 *)
4408                            <1>     ;    Load CS:IP from stack;
4409                            <1>     ;    Set CS RPL to CPL;
4410                            <1>     ;    Increment ESP by 4 plus the immediate offset if it exists;
4411                            <1>     ;    Load SS:ESP from stack;
4412                            <1>     ; FI;
4413                            <1>     ;
4414                            <1>     ; //
4415                            <1>
4416                            <1> int33h_2:
4417 00004FA4 80FC05            <1>     cmp     ah, 05h ; format track
4418 00004FA7 7709              <1>     ja      short int33h_3
4419 00004FA9 722B              <1>     jb      short int33h_5
4420 00004FAB 51                <1>     push    ecx
4421                            <1>     ;mov    ecx, 1
4422                            <1>     ; 17/07/2022
4423 00004FAC 31C9              <1>     xor     ecx, ecx
4424 00004FAE FEC1              <1>     inc     cl
4425                            <1>     ; ecx = 1
4426 00004FB0 EBC9              <1>     jmp     short int33h_1
4427                            <1> int33h_3:
4428 00004FB2 80FC1C            <1>     cmp     ah, 1Ch ; LBA write
4429 00004FB5 771F              <1>     ja      short int33h_5
4430 00004FB7 74B8              <1>     je      short int33h_0
4431 00004FB9 80FC1B            <1>     cmp     ah, 1Bh ; LBA read
4432 00004FBC 740D              <1>     je      short int33h_4
4433 00004FBE 80FC08            <1>     cmp     ah, 08h ; get disk parameters
4434 00004FC1 7513              <1>     jne     short int33h_5
4435                            <1>     ; 01/06/2016
```

```
4436 00004FC3 8B1D[0C830100]      <1>     mov     ebx, [user_buffer] ; user's buffer address
4437 00004FC9 EB10                <1>     jmp     short int33h_6
4438                              <1> int33h_4:
4439                              <1>     ;; 17/07/2022 - 64K r/w buffer limit check ?
4440                              <1>     ;cmp    al, 80h ; 128
4441                              <1>     ;ja     short int33h_8 ; error
4442                              <1>     ;; 17/07/2022 - zero r/w count check ?
4443                              <1>     ;or     al, al
4444                              <1>     ;jz     short int33h_8 ; error
4445                              <1>
4446                              <1>     ; 17/07/2022 (buffer limit and zero count check)
4447 00004FCB FEC8                <1>     dec     al
4448 00004FCD 78C0                <1>     js      short int33h_8 ; error
4449 00004FCF FEC0                <1>     inc     al
4450                              <1>
4451 00004FD1 A2[467C0100]        <1>     mov     byte [scount], al ; <= 128 sectors
4452                              <1> int33h_5:
4453 00004FD6 BB00000700          <1>     mov     ebx, Cluster_Buffer ; max. 65536 bytes
4454                              <1>                                 ; buf. addr: 70000h
4455                              <1>     ;mov    byte [ClusterBuffer_Valid], 0
4456                              <1> int33h_6:
4457                              <1>     ; 13/07/2022
4458                              <1>     ;pop    ds
4459                              <1>     ;pushfd
4460                              <1>     ;push   cs
4461                              <1>
4462 00004FDB E83B000000          <1>     call    DISK_IO
4463                              <1>
4464                              <1>     ;;mov   ebx, [cs:user_buffer] ; 01/06/2016
4465                              <1>     ;mov    ebx, [user_buffer] ; 13/07/2022
4466 00004FE0 72B2                <1>     jc      short int33h_9
4467                              <1>     ;
4468                              <1>     ;cmp    byte [cs:scount], 0
4469                              <1>     ;jna    short int33h_7
4470 00004FE2 803D[467C0100]00    <1>     cmp     byte [scount], 0 ; 13/07/2022
4471 00004FE9 76A9                <1>     jna     short int33h_9
4472                              <1>
4473                              <1>     ; 13/07/2022
4474                              <1>
4475                              <1>     ; transfer sector buffer content to user's buffer
4476                              <1>     ;push   es
4477                              <1>     ;push   ds
4478                              <1>     ;push   eax
4479                              <1>     ;mov    ax, KDATA
4480                              <1>     ;mov    ds, ax
4481                              <1>     ;mov    es, ax
4482                              <1>     ;push   ecx
4483                              <1>     ;push   esi
4484                              <1>     ;push   edi
4485                              <1>
4486                              <1>     ; 13/07/2022
4487 00004FEB 50                  <1>     push    eax
4488 00004FEC 0FB60D[467C0100]    <1>     movzx   ecx, byte [scount]
4489 00004FF3 C1E109              <1>     shl     ecx, 9 ; * 512 bytes
4490                              <1>     ;mov    edi, ebx ; user's buffer address
4491 00004FF6 8B3D[0C830100]      <1>     mov     edi, [user_buffer] ; 13/07/2022
4492 00004FFC BE00000700          <1>     mov     esi, Cluster_Buffer
4493 00005001 E8E4BA0000          <1>     call    transfer_to_user_buffer
4494                              <1>             ; (ecx and eax will be modified)
4495 00005006 58                  <1>     pop     eax
4496                              <1>
4497                              <1>     ; 13/07/2022
4498 00005007 7286                <1>     jc      short int33h_8 ; eax = 0FFh
4499 00005009 EB89                <1>     jmp     short int33h_9 ; cf = 0
4500                              <1>
4501                              <1>     ;pop    edi
4502                              <1>     ;pop    esi
4503                              <1>     ;pop    ecx
4504                              <1>     ;pop    eax
4505                              <1>     ;pop    ds
4506                              <1>     ;pop    es
4507                              <1>     ;jc     short int33h_8
4508                              <1> ;int33h_7:
4509                              <1>     ;cli
4510                              <1>     ;;;15/01/2017
4511                              <1>     ;;;mov  byte [ss:intflg], 0 ; 07/01/2017
4512                              <1>     ;; cf = 0  ; use eflags which is in stack
4513                              <1>     ;iretd
4514                              <1> ;int33h_8:
4515                              <1>     ;mov    eax, 0FFh ; Unknown error !?
4516                              <1>     ; 13/07/2022
4517                              <1>     ;xor    eax, eax
4518                              <1>     ;dec    al  ; eax = 0FFh
4519                              <1>     ;jmp    short int33h_9
4520                              <1>
4521                              <1> ;int33h_9:
4522                              <1>     ;; cf = 1
4523                              <1>     ;
4524                              <1>     ;; (*) 29/05/2016
4525                              <1>     ;; (*) retf 4 ; skip eflags on stack
4526                              <1>     ;; Note: This 'retf 4' was wrong, -it was causing
4527                              <1>     ;;       to stack errors in ring 3-
4528                              <1>     ;;     POP sequence of 'retf 4' is as
4529                              <1>     ;;       "eip, cs, eflags, esp, ss, +4 bytes"
4530                              <1>        ;;       it is not as "eip, cs, +4 bytes, esp, ss" !
4531                              <1>     ;
4532                              <1>     ;; 29/05/2016 -set carry flag on stack-
4533                              <1>     ;or     byte [esp+8], 1  ; set carry bit of eflags register
4534                              <1>     ;;iretd
4535                              <1>     ;jmp    short int33h_7 ; 07/01/2017
4536                              <1>
4537                              <1> ;; 11/04/2021
4538                              <1> ;int13h: ; 21/02/2015
4539                              <1> ;   clc ; 11/04/2021
4540                              <1> ;   pushfd
4541                              <1> ;   push    cs
4542                              <1> ;   call    DISK_IO
4543                              <1> ;   retn
4544                              <1>
4545                              <1> int13h:
4546                              <1>     ; 13/07/2022 - TRDOS 386 v2.0.5
4547                              <1>     ; Note: DISK_IO sets registers on stack
4548                              <1>     ;       as return parameters. So,
4549                              <1>     ;       stack order (at the entry of 'DISK_IO')
4550                              <1>     ;       must be same with 'int33h:' as above.
4551                              <1>
4552                              <1>     ;push   es ; not necessary
4553                              <1>     ;push   ds ; not necessary
4554                              <1>     ;
4555                              <1>     ; following pushes are necessary
4556                              <1>     ; for setting registers -return values- in DISK_IO
4557 0000500B 53                  <1>     push    ebx
4558 0000500C 51                  <1>     push    ecx
4559 0000500D 52                  <1>     push    edx
```

```
4560 0000500E 56                <1>     push    esi
4561 0000500F 57                <1>     push    edi
4562                            <1>     ;push   ebp
4563                            <1>     ;mov    ebp, esp
4564                            <1>     ; edi = ebp+4
4565                            <1>     ; esi = ebp+8
4566                            <1>     ; edx = ebp+12
4567                            <1>     ; ecx = ebp+16
4568                            <1>     ; ebx = ebp+20
4569                            <1>     ;
4570 00005010 E806000000        <1>     call    DISK_IO
4571                            <1>     ;
4572 00005015 5F                <1>     pop     edi
4573 00005016 5E                <1>     pop     esi
4574 00005017 5A                <1>     pop     edx
4575 00005018 59                <1>     pop     ecx
4576 00005019 5B                <1>     pop     ebx
4577                            <1>     ;
4578                            <1>     ;pop    ds
4579                            <1>     ;pop    es
4580                            <1>     ;
4581 0000501A C3                <1>     retn
4582                            <1>
4583                            <1> ; 10/08/2022
4584                            <1> ; 07/08/2022
4585                            <1> ; 17/07/2022
4586                            <1> ; 13/07/2022 - TRDOS 386 v2.0.5
4587                            <1> ; 18/04/2021 - TRDOS 386 v2.0.4
4588                            <1> ; 11/04/2021 - TRDOS 386 v2.0.3
4589                            <1> ; 30/08/2020
4590                            <1> ; 09/12/2017
4591                            <1> ; 29/05/2016
4592                            <1> ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
4593                            <1>
4594                            <1> DISK_IO:
4595                            <1>     ; 10/08/2022
4596                            <1>     ; 17/07/2022
4597                            <1>     ; 13/07/2022
4598                            <1>     ; Registers are also on stack
4599                            <1>     ; (with same contents)
4600                            <1>     ; in following order:
4601                            <1>     ;
4602                            <1>     ;    ebx = esp+20
4603                            <1>     ;    ecx = esp+16
4604                            <1>     ;    edx = esp+12
4605                            <1>     ;    esi = esp+8
4606                            <1>     ;    edi = esp+4
4607                            <1>
4608                            <1>     ; cs = KCODE (== KDATA base address)
4609                            <1>     ; ss = KDATA
4610                            <1>     ; ds = KDATA
4611                            <1>     ; es = KDATA
4612                            <1>
4613                            <1>     ; 17/07/2022
4614 0000501B FB                <1>     sti                             ; ENABLE INTERRUPTS
4615                            <1>
4616 0000501C 80FA80            <1>     cmp     dl, 80h                 ; TEST FOR FIXED DISK DRIVE
4617                            <1>     ;jae    short A1                ; YES, HANDLE HERE
4618                            <1>     ;;;int  40h                     ; DISKETTE HANDLER
4619                            <1>     ;;call  int40h
4620                            <1>     ;jb     DISKETTE_IO_1
4621                            <1> ;RET_2:
4622                            <1>     ;;retf  2                       ; BACK TO CALLER
4623                            <1>     ;retf   4
4624                            <1>     ; 11/04/2021
4625 0000501F 7305              <1>     jnb     short A1
4626 00005021 E934F2FFFF        <1>     jmp     DISKETTE_IO_1
4627                            <1> A1:
4628                            <1>     ;sti     ; 17/07/2022           ; ENABLE INTERRUPTS
4629                            <1>
4630                            <1>     ;; 04/01/2015
4631                            <1>     ;;or    ah, ah
4632                            <1>     ;;jnz   short A2
4633                            <1>     ;;int   40h                     ; RESET NEC WHEN AH=0
4634                            <1>     ;;sub   ah, ah
4635                            <1>
4636 00005026 80FA83            <1>     cmp     dl, (80h + S_MAX_FILE - 1)
4637                            <1>     ;ja     short RET_2
4638 00005029 760A              <1>     jna     short _A0
4639                            <1>
4640                            <1>     ; 13/07/2022
4641                            <1>     ; (here, DS is KDATA segment already)
4642                            <1>     ;
4643                            <1>     ; 29/05/2016
4644                            <1>     ;push   ds
4645                            <1>     ; 11/04/2021
4646                            <1>     ;push   eax
4647                            <1>     ;mov    ax, KDATA
4648                            <1>     ;mov    ds, ax
4649                            <1>     ; 11/04/2021
4650                            <1>     ;pop    eax
4651                            <1>
4652 0000502B B4AA              <1>     mov     ah, 0AAh        ; Hard disk drive not ready !
4653                            <1>                             ; (Programmer's guide to AMIBIOS, 1992)
4654 0000502D 8825[17770100]    <1>     mov     byte [DISK_STATUS1], ah
4655                            <1>     ; 13/07/2022
4656                            <1>     ;pop    ds
4657                            <1>     ;jmp    short RET_2
4658 00005033 F9                <1>     stc
4659 00005034 C3                <1>     retn
4660                            <1> _A0:
4661                            <1>     ; 18/01/2015
4662 00005035 08E4              <1>     or      ah, ah
4663 00005037 742C              <1>     jz      short A4
4664 00005039 80FC0D            <1>     cmp     ah, 0Dh ; Alternate reset
4665 0000503C 7504              <1>     jne     short A2
4666 0000503E 28E4              <1>     sub     ah, ah  ; Reset
4667 00005040 EB23              <1>     jmp     short A4
4668                            <1> A2:
4669                            <1>     ; 13/07/2022
4670 00005042 80FC08            <1>     cmp     ah, 08h                 ; GET PARAMETERS IS A SPECIAL CASE
4671 00005045 7505              <1>     jne     short A3
4672 00005047 E911040000        <1>         jmp GET_PARM_N
4673                            <1> A3:
4674                            <1>     ; 13/07/2022
4675 0000504C 80FC15            <1>     cmp     ah, 15h                 ; READ DASD TYPE IS ALSO
4676 0000504F 7514              <1>     jne     short A4
4677 00005051 E9AC030000        <1>         jmp READ_DASD_TYPE ; Return Drive Type
4678                            <1>                            ; (Programmer's guide to AMIBIOS, 1992)
4679                            <1>     ; 13/07/2022
4680                            <1> int33h_bad_cmd:
4681                            <1>     ; 16/05/2016
4682                            <1>     ; 30/01/2015
4683                            <1>     ; 29/05/2016
```

```
                            <1>     ;push  ds
                            <1>     ;push  eax
                            <1>     ;mov   ax, KDATA
                            <1>     ;mov   ds, ax
                            <1>     ;pop   eax
                            <1>
00005056 B401               <1>     mov    ah, BAD_CMD
00005058 8825[17770100]     <1>     mov     [DISK_STATUS1], ah ; BAD_CMD  ; COMMAND ERROR
                            <1>        ;jmp       short RET_2
                            <1>     ; 13/07/2022
                            <1> ;RET_2:
                            <1>     ; (*) 29/05/2016
                            <1>     ; (*) retf 4
                            <1>     ;or    byte [esp+8], 1 ; set carry bit of eflags register
                            <1>     ;iretd
                            <1>
                            <1>     ; 13/07/2022
0000505E F9                 <1>     stc
                            <1>     ; cf = 1, ah = BAD_CMD
0000505F C3                 <1>     retn
                            <1> _A4:
                            <1>     ; 13/07/2022
                            <1>     ; 02/02/2015
00005060 80FC1D             <1>     cmp    ah, 1Dh                ; (Temporary for Retro UNIX 386 v1)
                            <1>     ; 12/01/2015
                            <1>     ;cmc
                            <1>     ;jnc   short A4
                            <1>     ; 13/07/2022
00005063 73F1               <1>     jnb    short int33h_bad_cmd
                            <1> A4:                                ; SAVE REGISTERS DURING OPERATION
00005065 C8080000           <1>     enter  8, 0                    ; SAVE (BP) AND MAKE ROOM FOR @CMD_BLOCK
                            <1>
                            <1>     ; 13/07/2022
                            <1>     ; ENTER 8, 0
                            <1>     ;;push ebp
                            <1>     ;;mov  ebp, esp
                            <1>     ;;sub  esp, 8
                            <1>     ;
                            <1>     ;push  ebx                     ;  IN THE STACK, THE COMMAND BLOCK IS:
                            <1>     ;push  ecx                     ;   @CMD_BLOCK == BYTE PTR [BP]-8
                            <1>     ;push  edx
                            <1>     ;push  esi
                            <1>     ;push  edi
                            <1>
                            <1>     ; 13/07/2022
                            <1>     ; edi = ebp+8
                            <1>     ; esi = ebp+12
                            <1>     ; edx = ebp+16
                            <1>     ; ecx = ebp+20
                            <1>     ; ebx = ebp+24
                            <1>
                            <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>     ; 13/07/2022
00005069 E880000000         <1>     call   DISK_IO_CONT            ; PERFORM THE OPERATION
                            <1>     ;;call DDS                     ; ESTABLISH SEGMENT
0000506E 8A25[17770100]     <1>     mov    ah, [DISK_STATUS1]      ; GET STATUS FROM OPERATION
                            <1>     ;(*) cmp ah, 1                 ; SET THE CARRY FLAG TO INDICATE
                            <1>                                    ; SUCCESS OR FAILURE
                            <1>     ;pop   edi                     ; RESTORE REGISTERS
                            <1>     ;pop   esi
                            <1>     ;pop   edx
                            <1>     ;pop   ecx
                            <1>     ;pop   ebx
                            <1>
00005074 C9                 <1>     leave                          ; ADJUST (SP) AND RESTORE (BP)
                            <1>
                            <1>     ;retf  2                       ; THROW AWAY SAVED FLAGS
                            <1>     ; (*) 29/05/2016
                            <1>     ; (*) retf 4
                            <1>
                            <1>     ; 13/07/2022
00005075 80FC01             <1>     cmp    ah, 1
                            <1>     ;jc    short _A5
                            <1>     ;or    byte [esp+8], 1 ; set carry bit of eflags register
                            <1> ;_A5:
                            <1>     ;iretd
                            <1>     ; 10/08/2022
00005078 F5                 <1>     cmc
                            <1>     ; 13/07/2022
00005079 C3                 <1>     retn
                            <1>
                            <1> ; 21/02/2015
                            <1> ;       dw --> dd
                            <1>     ; 13/07/2022
                            <1> D1:                                ; FUNCTION TRANSFER TABLE
0000507A [91520000]         <1>     dd     DISK_RESET              ; 00h
0000507E [F1520000]         <1>     dd     RETURN_STATUS           ; 01h
00005082 [04530000]         <1>     dd     DISK_READ               ; 02h
00005086 [60530000]         <1>     dd     DISK_WRITE              ; 03h
0000508A [EA530000]         <1>     dd     DISK_VERF               ; 04h
0000508E [D4530000]         <1>     dd     FMT_TRK                 ; 05h
00005092 [27520000]         <1>     dd     BAD_COMMAND             ; 06h  FORMAT BAD SECTORS
00005096 [27520000]         <1>     dd     BAD_COMMAND             ; 07h  FORMAT DRIVE
0000509A [27520000]         <1>     dd     BAD_COMMAND             ; 08h  RETURN PARAMETERS
0000509E [DD540000]         <1>     dd     INIT_DRV                ; 09h
000050A2 [FE520000]         <1>     dd     RD_LONG                 ; 0Ah
000050A6 [5A530000]         <1>     dd     WR_LONG                 ; 0Bh
000050AA [4E550000]         <1>     dd     DISK_SEEK               ; 0Ch
000050AE [91520000]         <1>     dd     DISK_RESET              ; 0Dh
000050B2 [27520000]         <1>     dd     BAD_COMMAND             ; 0Eh  READ BUFFER
000050B6 [27520000]         <1>     dd     BAD_COMMAND             ; 0Fh  WRITE BUFFER
000050BA [76550000]         <1>     dd     TST_RDY                 ; 10h
000050BE [AA550000]         <1>     dd     HDISK_RECAL             ; 11h
000050C2 [27520000]         <1>     dd     BAD_COMMAND             ; 12h  MEMORY DIAGNOSTIC
000050C6 [27520000]         <1>     dd     BAD_COMMAND             ; 13h  DRIVE DIAGNOSTIC
000050CA [E0550000]         <1>     dd     CTLR_DIAGNOSTIC         ; 14h  CONTROLLER DIAGNOSTIC
                            <1>     ;; 02/02/2015 (Temporary - Retro UNIX 386 v1 - DISK I/O test)
000050CE [27520000]         <1>     dd     BAD_COMMAND             ; 15h
000050D2 [27520000]         <1>     dd     BAD_COMMAND             ; 16h
000050D6 [27520000]         <1>     dd     BAD_COMMAND             ; 17h
000050DA [27520000]         <1>     dd     BAD_COMMAND             ; 18h
000050DE [27520000]         <1>     dd     BAD_COMMAND             ; 19h
000050E2 [27520000]         <1>     dd     BAD_COMMAND             ; 1Ah
000050E6 [04530000]         <1>     dd     DISK_READ               ; 1Bh ; LBA read
000050EA [60530000]         <1>     dd     DISK_WRITE              ; 1Ch ; LBA write
                            <1> D1L     EQU    $ - D1
                            <1>
                            <1>     ; 02/12/2023
                            <1>     ; 01/12/2023 - TRDOS 386 v2.0.7
                            <1>     ; 07/08/2022
```

```
                            <1>     ; 17/07/2022 - TRDOS 386 v2.0.5
                            <1> DISK_IO_CONT:
                            <1>     ;;call DDS                      ; ESTABLISH SEGMENT
                            <1>     ; 11/04/2021
000050EE 80FC01             <1>     cmp     ah, 01h                 ; RETURN STATUS
000050F1 7505               <1>     jne     short SU0
000050F3 E9F9010000         <1>     jmp     RETURN_STATUS
                            <1> SU0:
000050F8 C605[17770100]00   <1>     mov     byte [DISK_STATUS1], 0        ; RESET THE STATUS INDICATOR
                            <1>     ; 13/07/2022
000050FF 89DE               <1>     mov     esi, ebx ; 21/02/2015 ; SAVE DATA ADDRESS
00005101 8A1D[18770100]     <1>     mov     bl, [HF_NUM]            ; GET NUMBER OF DRIVES
00005107 80E27F             <1>     and     dl, 7Fh                 ; GET DRIVE AS 0 OR 1
                            <1>                                     ; (get drive number as 0 to 3)
                            <1>     ; 14/02/2015
0000510A 38D3               <1>     cmp     bl, dl
                            <1>     ;jbe    short BAD_COMMAND       ; INVALID DRIVE
                            <1>     ; 07/08/2022
0000510C 7705               <1>     ja      short SU0X
0000510E E914010000         <1>     jmp     BAD_COMMAND
                            <1> SU0X:
                            <1>     ;;03/01/2015
00005113 29DB               <1>     sub     ebx, ebx
00005115 88D3               <1>     mov     bl, dl
00005117 883D[2C770100]     <1>     mov     [LBAMode], bh  ; 0
                            <1>
                            <1>     ;test   byte [ebx+hd0_type], 1        ; LBA ready ?
                            <1>     ;jz     short su1               ; no
                            <1>     ;inc    byte [LBAMode]
                            <1> ;su1:
                            <1>     ; 11/04/2021 (32 bit push/pop)
                            <1>     ; 21/02/2015 (32 bit modification)
                            <1>     ; 04/01/2015
0000511D 50                 <1>     push    eax ; ***
                            <1>     ;push   es  ; **
0000511E 52                 <1>     push    edx ; *
0000511F 50                 <1>     push    eax ; ****
00005120 E8AA060000         <1>     call    GET_VEC                 ; GET DISK PARAMETERS
                            <1>     ; 02/02/2015
                            <1>     ;mov    ax, [ES:BX+16] ; I/O port base address (1F0h, 170h)
00005125 668B4310           <1>     mov     ax, [ebx+16]
00005129 66A3[6E660000]     <1>     mov     [HF_PORT], ax
                            <1>     ;mov    dx, [ES:BX+18] ; control port address (3F6h, 376h)
0000512F 668B5312           <1>     mov     dx, [ebx+18]
00005133 668915[70660000]   <1>     mov     [HF_REG_PORT], dx
                            <1>     ;mov    al, [ES:BX+20] ; head register upper nibble (A0h,B0h,E0h,F0h)
0000513A 8A4314             <1>     mov     al, [ebx+20]
                            <1>     ; 23/02/2015
0000513D A840               <1>     test    al, 40h  ; LBA bit (bit 6)
0000513F 7406               <1>     jz      short su1
00005141 FE05[2C770100]     <1>     inc     byte [LBAMode] ; 1
                            <1> su1:
00005147 C0E804             <1>     shr     al, 4
0000514A 2401               <1>     and     al, 1
0000514C A2[72660000]       <1>     mov     [hf_m_s], al
                            <1>     ;
                            <1>     ; 03/01/2015
                            <1>     ;mov    al, [ES:BX+8]           ; GET CONTROL BYTE MODIFIER
00005151 8A4308             <1>     mov     al, [ebx+8]
                            <1>     ;mov    dx, [HF_REG_PORT]       ; Device Control register
00005154 EE                 <1>     out     dx, al                  ; SET EXTRA HEAD OPTION
                            <1>                                     ; -here-
                            <1>                                     ; Control Byte: (= 08h)
                            <1>                                     ;  bit 0 - 0
                            <1>                                     ;  bit 1 - nIEN (1 = disable irq)
                            <1>                                     ;  bit 2 - SRST (software RESET)
                            <1>                                     ;  bit 3 - use extra heads (8 to 15)
                            <1>                                     ;          -always set to 1-
                            <1>                                     ;  (bits 3 to 7 are reserved
                            <1>                                     ;          for ATA devices)
00005155 8A25[19770100]     <1>     mov     ah, [CONTROL_BYTE]      ; SET EXTRA HEAD OPTION IN
0000515B 80E4C0             <1>     and     ah, 0C0h                ; CONTROL BYTE
0000515E 08C4               <1>     or      ah, al
00005160 8825[19770100]     <1>     mov     [CONTROL_BYTE], ah
                            <1>
                            <1>     ; 11/04/2021 (32 bit push/pop)
                            <1>     ; 04/01/2015
00005166 58                 <1>     pop     eax ; ****
00005167 5A                 <1>     pop     edx ; * ; 14/02/2015
00005168 20E4               <1>     and     ah, ah  ; Reset function ?
0000516A 7506               <1>     jnz     short su2
                            <1>     ;pop    es ; **
0000516C 58                 <1>     pop     eax ; ***
0000516D E91F010000         <1>         jmp     DISK_RESET
                            <1> su2:
00005172 803D[2C770100]00   <1>     cmp     byte [LBAMode], 0
00005179 765E               <1>     jna     short su3
                            <1>     ;
                            <1>     ; 02/02/2015 (LBA read/write function calls)
0000517B 80FC1B             <1>     cmp     ah, 1Bh
0000517E 720A               <1>     jb      short lbarw1
00005180 80FC1C             <1>     cmp     ah, 1Ch
00005183 7759               <1>     ja      short invldfnc
                            <1>     ;;pop   edx ; * ; 14/02/2015
                            <1>     ;mov    ax, cx ; Lower word of LBA address (bits 0-15)
                            <1>
                            <1>     ; 01/12/2023 (48 bit LBA rw)
00005185 E9AD000000         <1>     jmp     lba_read_write
                            <1>
                            <1> lbarw1:
                            <1>     ; convert CHS to LBA
                            <1>     ;
                            <1>     ; LBA calculation - AWARD BIOS - 1999 - AHDSK.ASM
                            <1>     ; LBA = "# of Heads" * Sectors/Track * Cylinder + Head * Sectors/Track
                            <1>     ;       + Sector - 1
                            <1>     ; 11/04/2021 (32 bit push/pop)
0000518A 52                 <1>     push    edx ; * ;; 14/02/2015
                            <1>     ;xor    dh, dh
0000518B 31D2               <1>     xor     edx, edx
                            <1>     ;mov    dl, [ES:BX+14] ; sectors per track (logical)
0000518D 8A530E             <1>     mov     dl, [ebx+14]
                            <1>     ;xor    ah, ah
00005190 31C0               <1>     xor     eax, eax
                            <1>     ;mov    al, [ES:BX+2]  ; heads (logical)
00005192 8A4302             <1>     mov     al, [ebx+2]
00005195 FEC8               <1>     dec     al
00005197 6640               <1>     inc     ax              ; 0 = 256
00005199 66F7E2             <1>     mul     dx
                            <1>             ; AX = # of Heads * Sectors/Track
0000519C 6689CA             <1>     mov     dx, cx
                            <1>     ;and    cx, 3Fh  ; sector  (1 to 63)
0000519F 83E13F             <1>     and     ecx, 3Fh
000051A2 86F2               <1>     xchg    dl, dh
000051A4 C0EE06             <1>     shr     dh, 6
```

```
                                <1>             ; DX = cylinder (0 to 1023)
                                <1>     ;mul    dx
                                <1>             ; DX:AX = # of Heads" * Sectors/Track * Cylinder
000051A7 F7E2                   <1>     mul     edx
000051A9 FEC9                   <1>     dec     cl ; sector - 1
                                <1>     ;add    ax, cx
                                <1>     ;adc    dx, 0
                                <1>             ; DX:AX = # of Heads" * Sectors/Track * Cylinder + Sector - 1
000051AB 01C8                   <1>     add     eax, ecx
                                <1>     ; 11/04/2021 (32 bit push/pop)
000051AD 59                     <1>     pop     ecx ; * ; ch = head, cl = drive number (zero based)
                                <1>     ;push   dx
                                <1>     ;push   ax
000051AE 50                     <1>     push    eax
                                <1>     ; 13/07/2022
000051AF 29C0                   <1>     sub     eax, eax
                                <1>     ;mov    al, [ES:BX+14] ; sectors per track (logical)
000051B1 8A430E                 <1>     mov     al, [ebx+14]
000051B4 F6E5                   <1>     mul     ch
                                <1>             ; AX = Head * Sectors/Track
                                <1>         ; 13/07/2022
                                <1>     ;movzx  eax, ax ; 09/12/2017
                                <1>     ;pop    dx
000051B6 5A                     <1>     pop     edx
                                <1>     ;add    ax, dx
                                <1>     ;pop    dx
                                <1>     ;adc    dx, 0 ; add carry bit
000051B7 01D0                   <1>     add     eax, edx
                                <1> lbarw2:
000051B9 29D2                   <1>     sub     edx, edx ; 21/02/2015
000051BB 88CA                   <1>     mov     dl, cl ; 21/02/2015
000051BD C645F800               <1>         mov     byte [CMD_BLOCK], 0 ; Features Register
                                <1>                             ; NOTE: Features register (1F1h, 171h)
                                <1>                             ; is not used for ATA device R/W functions.
                                <1>                             ; It is old/obsolete 'write precompensation'
                                <1>                             ; register and error register
                                <1>                             ; for old ATA/IDE devices.
                                <1>     ; 18/01/2014
000051C1 8A0D[72660000]         <1>     ;mov    ch, [hf_m_s]   ; Drive 0 (master) or 1 (slave)
                                <1>     mov     cl, [hf_m_s]
                                <1>     ;shl    ch, 4           ; bit 4 (drive bit)
                                <1>     ;or     ch, 0E0h        ; bit 5 = 1
                                <1>                             ; bit 6 = 1 = LBA mode
                                <1>                             ; bit 7 = 1
000051C7 80C90E                 <1>     or      cl, 0Eh ; 1110b
                                <1>     ;and    dh, 0Fh         ; LBA byte 4 (bits 24 to 27)
000051CA 25FFFFFF0F             <1>     and     eax, 0FFFFFFFh
000051CF C1E11C                 <1>     shl     ecx, 28 ; 21/02/2015
                                <1>     ;or     dh, ch
000051D2 09C8                   <1>     or      eax, ecx
                                <1>     ;;mov   [CMD_BLOCK+2], al ; LBA byte 1 (bits 0 to 7)
                                <1>                               ; (Sector Number Register)
                                <1>     ;;mov   [CMD_BLOCK+3], ah ; LBA byte 2 (bits 8 to 15)
                                <1>                               ; (Cylinder Low Register)
                                <1>     ;mov    [CMD_BLOCK+2], ax ; LBA byte 1, 2
                                <1>     ;mov    [CMD_BLOCK+4], dl ; LBA byte 3 (bits 16 to 23)
                                <1>                               ; (Cylinder High Register)
                                <1>     ;;mov   [CMD_BLOCK+5], dh ; LBA byte 4 (bits 24 to 27)
                                <1>                               ; (Drive/Head Register)
                                <1>
                                <1>     ;mov    [CMD_BLOCK+4], dx ; LBA byte 4, LBA & DEV select bits
000051D4 8945FA                 <1>     mov     [CMD_BLOCK+2], eax ; 21/02/2015
                                <1>     ; 14/02/2015
                                <1>     ;mov    dl, cl ; Drive number (INIT_DRV)
000051D7 EB36                   <1>     jmp     short su4
                                <1> su3:
                                <1>     ; 07/08/2022
                                <1>     ; 13/07/2022
                                <1>     ; 02/02/2015
                                <1>     ; (1Bh & 1Ch functions are not valid for CHS mode)
000051D9 80FC14                 <1>     cmp     ah, 14h
000051DC 7603                   <1>     jna     short chsfnc
                                <1> invldfnc:
                                <1>         ; 14/02/2015
                                <1>     ;pop    es ; **
                                <1>     ; 11/04/2021
000051DE 58                     <1>     pop     eax ; ***
000051DF EB46                   <1>         jmp     short BAD_COMMAND
                                <1> chsfnc:
                                <1>     ;mov    ax, [ES:BX+5]         ; GET WRITE PRE-COMPENSATION CYLINDER
000051E1 668B4305               <1>     mov     ax, [ebx+5]
                                <1>     ;shr    ax, 2
                                <1>     ; 07/08/2022
000051E5 C1E802                 <1>     shr     eax, 2
000051E8 8845F8                 <1>     mov     [CMD_BLOCK], al
                                <1>
                                <1>     ;;mov   al, [ES:BX+8]         ; GET CONTROL BYTE MODIFIER
                                <1>     ;;push  edx ; *
                                <1>     ;;mov   dx, [HF_REG_PORT]
                                <1>     ;;out   dx, al                ; SET EXTRA HEAD OPTION
                                <1>     ;;pop   edx ; *
                                <1>     ;;pop   es  ; **
                                <1>     ;;mov   ah, [CONTROL_BYTE]    ; SET EXTRA HEAD OPTION IN
                                <1>     ;;and   ah, 0C0h              ; CONTROL BYTE
                                <1>     ;;or    ah, al
                                <1>     ;;mov   [CONTROL_BYTE], ah
                                <1>
000051EB 88C8                   <1>     mov     al, cl                ; GET SECTOR NUMBER
000051ED 243F                   <1>     and     al, 3Fh
000051EF 8845FA                 <1>     mov     [CMD_BLOCK+2], al
000051F2 886DFB                 <1>     mov     [CMD_BLOCK+3], ch     ; GET CYLINDER NUMBER
000051F5 88C8                   <1>     mov     al, cl
000051F7 C0E806                 <1>     shr     al, 6
000051FA 8845FC                 <1>     mov     [CMD_BLOCK+4], al     ; CYLINDER HIGH ORDER 2 BITS
                                <1>
                                <1>     ;;05/01/2015
                                <1>     ;;mov   al, dl                ; DRIVE NUMBER
000051FD A0[72660000]           <1>     mov     al, [hf_m_s]
00005202 C0E004                 <1>     shl     al, 4
00005205 80E60F                 <1>     and     dh, 0Fh               ; HEAD NUMBER
00005208 08F0                   <1>     or      al, dh
                                <1>     ;or     al, 80h or 20h
0000520A 0CA0                   <1>     or      al, 80h+20h           ; ECC AND 512 BYTE SECTORS
0000520C 8845FD                 <1>     mov     [CMD_BLOCK+5], al     ; ECC/SIZE/DRIVE/HEAD
                                <1> su4:
                                <1>     ;pop    es ; **
                                <1>         ;; 14/02/2015
                                <1>         ;;pop      ax
                                <1>         ;;mov      [CMD_BLOCK+1], al     ; SECTOR COUNT
                                <1>         ;;push     ax
                                <1>         ;;mov      al, ah                ; GET INTO LOW BYTE
                                <1>         ;;xor      ah, ah                ; ZERO HIGH BYTE
                                <1>         ;;sal      ax, 1                 ; *2 FOR TABLE LOOKUP
                                <1>         ; 11/04/2021
```

```
5056 0000520F 58                <1>     pop     eax ; ***
5057 00005210 8845F9            <1>         mov     [CMD_BLOCK+1], al
5058 00005213 29DB              <1>         sub ebx, ebx
5059                            <1>     ;xor    bh, bh
5060 00005215 88E3              <1>     mov     bl, ah
5061                            <1>         ;sal       bx, 1
5062                            <1>         ; 17/07/2022
5063 00005217 C1E302            <1>     sal     ebx, 2  ; 32 bit offset (21/02/2015)
5064                            <1>     ;mov    si, ax                  ; PUT INTO SI FOR BRANCH
5065                            <1>         ; 13/07/2022
5066                            <1>     ;cmp    bx, D1L                 ; TEST WITHIN RANGE
5067                            <1>     ;jnb    short BAD_COMMAND_POP
5068 0000521A 83FB74            <1>     cmp     ebx, D1L                ; TEST WITHIN RANGE
5069 0000521D 7308              <1>     jnb     short BAD_COMMAND
5070                            <1>         ;xchg      bx, si
5071 0000521F 87F3              <1>         xchg       ebx, esi
5072                            <1>     ;;;pop  ax                      ; RESTORE AX
5073                            <1>     ;;;pop  bx                      ; AND DATA ADDRESS
5074                            <1>
5075                            <1>     ;;push  cx
5076                            <1>     ;;push  ax                      ; ADJUST ES:BX
5077                            <1>     ;mov    cx, bx                  ; GET 3 HIGH ORDER NIBBLES OF BX
5078                            <1>     ;shr    cx, 4
5079                            <1>     ;mov    ax, es
5080                            <1>     ;add    ax, cx
5081                            <1>     ;mov    es, ax
5082                            <1>     ;and    bx, 000Fh               ; ES:BX CHANGED TO ES:000X
5083                            <1>     ;;pop   ax
5084                            <1>     ;;pop   cx
5085                            <1>     ;;jmp   word [CS:SI+D1]
5086                            <1>     ;jmp    word [SI+D1]
5087                            <1>
5088 00005221 FFA6[7A500000]    <1>     jmp     dword [esi+D1]
5089                            <1>
5090                            <1>     ; 07/08/2022
5091                            <1>     ; 13/07/2022
5092                            <1> BAD_COMMAND:
5093 00005227 C605[17770100]01  <1>         mov byte [DISK_STATUS1], BAD_CMD ; COMMAND ERROR
5094 0000522E B000              <1>     mov     al, 0
5095 00005230 C3                <1>     retn
5096                            <1>
5097                            <1> ; ---------------------------------------------------
5098                            <1> ; 48 bit LBA read/write
5099                            <1> ; ---------------------------------------------------
5100                            <1>
5101                            <1> su10:       ; 01/12/2023 ; 28 bit LBA r/w
5102 00005231 89C8              <1>     mov     eax, ecx ; LBA address (21/02/2015)
5103                            <1>     ; 13/07/2022
5104 00005233 88D1              <1>     mov     cl, dl ; 14/02/2015
5105 00005235 EB82              <1>     jmp     short lbarw2
5106                            <1>
5107                            <1> lba_read_write:
5108                            <1>     ; 02/12/2023
5109                            <1>     ; 01/12/2023 - TRDOS 386 v2.0.7 (48 bit LBA rw)
5110 00005237 81F9FFFFFF0F      <1>     cmp     ecx, 0FFFFFFFh
5111 0000523D 731B              <1>     jnb     short su6
5112                            <1>     ;movzx  eax, byte [esp] ; ***
5113 0000523F 31C0              <1>     xor     eax, eax
5114 00005241 8A0424            <1>     mov     al, [esp] ; *** ; sector count
5115 00005244 01C8              <1>     add     eax, ecx
5116 00005246 730B              <1>     jnc     short su5
5117 00005248 C605[17770100]17  <1>     mov     byte [DISK_STATUS1], ERR_INV_PARAMETER
5118                            <1>     ;jmp    short su7
5119                            <1>     ; 20/06/2024
5120 0000524F 58                <1>     pop     eax ; ***
5121 00005250 B000              <1>     mov     al, 0
5122 00005252 C3                <1>     retn
5123                            <1> su5:
5124 00005253 3DFFFFFF0F        <1>     cmp     eax, 0FFFFFFFh ; 28 bit limit
5125 00005258 76D7              <1>     jna     short su10
5126                            <1> su6:
5127                            <1>
5128                            <1> ; 20/06/2024 (TRDOS 386 v2.0.8)
5129                            <1> %if 0
5130                            <1>     ; 48 bit LBA r/w
5131                            <1>     mov     al, [hf_m_s]   ; (+!+) ; 02/12/2023
5132                            <1>     shl     al, 4
5133                            <1>     ; 02/12/2023
5134                            <1>     ;add    al, 0E0h
5135                            <1>     add     al, 40h
5136                            <1>     ;;add   al, 0Eh
5137                            <1>     ;add    al, 04h
5138                            <1>     ;shl    al, 4
5139                            <1>     mov     dx, [HF_PORT]
5140                            <1>     add     dl, 6   ; hd base port + 6
5141                            <1>     out     dx, al
5142                            <1>     ; 02/12/2023
5143                            <1>     mov     [CMD_BLOCK+6], al
5144                            <1>     inc     edx     ; hd base port + 7
5145                            <1>     in      al, dx
5146                            <1>     NEWIODELAY
5147                            <1>     ;test   al, 128         ; READY ?
5148                            <1>     and     al, 128
5149                            <1>     jz      short su8
5150                            <1>     in      al, dx
5151                            <1>     ;test   al, 128
5152                            <1>     and     al, 128
5153                            <1>     jz      short su8
5154                            <1>     mov     byte [DISK_STATUS1], TIME_OUT
5155                            <1> su7:
5156                            <1>     pop     eax ; ***
5157                            <1>     mov     al, 0
5158                            <1>     retn
5159                            <1> su8:
5160                            <1>     ;mov    dx, [HF_PORT]
5161                            <1>     ;inc    edx
5162                            <1>     ;inc    edx     ; hd base port + 2
5163                            <1>     sub     dl, 5   ; hd base port + 2
5164                            <1>     ;xor    al, al
5165                            <1>     out     dx, al  ; sector count hb (bits 8 to 15) = 0
5166                            <1>     inc     edx     ; hd base port + 3
5167                            <1>     mov     eax, ecx ; LBA disk sector address
5168                            <1>     rol     eax, 8
5169                            <1>     out     dx, al  ; LBA byte 4 (bits 24 to 31)
5170                            <1>     inc     edx     ; hd base port + 4
5171                            <1>     xor     al, al
5172                            <1>     out     dx, al  ; LBA byte 5 (bits 32 to 39) = 0
5173                            <1>     inc     edx     ; hd base port + 5
5174                            <1>     ;sub    al, al
5175                            <1>     out     dx, al  ; LBA byte 6 (bits 40 to 47) = 0
5176                            <1>
5177                            <1>     mov     al, [esp] ; ***
5178                            <1>     ; 02/12/2023
5179                            <1>     mov     [CMD_BLOCK+1], al
```

```
                        <1>
                        <1>     sub     dl, 3   ; hd base port + 2
                        <1>     out     dx, al  ; sector count lb (bits 0 to 7)
                        <1>     inc     edx     ; hd base port + 3
                        <1>     mov     eax, ecx ; LBA disk sector address
                        <1>     out     dx, al  ; LBA byte 1 (bits 0 to 7)
                        <1>     inc     edx     ; hd base port + 4
                        <1>     shr     eax, 8
                        <1>     out     dx, al  ; LBA byte 2 (bits 8 to 15)
                        <1>     inc     edx     ; hd base port + 5
                        <1>     shr     eax, 8
                        <1>     out     dx, al  ; LBA byte 3 (bits 16 to 23)
                        <1>
                        <1>     inc      edx    ; hd base port + 6
                        <1>
                        <1>     ; 02/12/2023 (not necessary) (+!+)
                        <1>     ;mov    al, [hf_m_s]
                        <1>     ;shl    al, 4
                        <1>     ;add    al, 40h
                        <1>     ;out    dx, al
                        <1>
                        <1>     pop     eax     ; ***
                        <1>
                        <1>     inc     edx     ; dx = hd base port + 7
                        <1>                     ;      command/status port
                        <1>     ;xchg   esi, ebx
                        <1>     ;mov    edi, ebx
                        <1>     mov     edi, esi ; sector buffer
                        <1>     cmp     ah, 1Ch
                        <1>     jne     short su9
                        <1>     mov     al, 34h ; WRITE SECTOR(S) EXT
                        <1>     out     dx, al
                        <1>     jmp     CMD_WX
                        <1> su9:
                        <1>     mov     al, 24h ; READ SECTOR(S) EXT
                        <1>     out     dx, al
                        <1>     jmp     CMD_RX
                        <1> %else
                        <1>     ; 20/06/2024
0000525A 58             <1>     pop     eax     ; ***
0000525B C645F800       <1>     mov     byte [CMD_BLOCK], 0
0000525F 8845F9         <1>     mov     [CMD_BLOCK+1], al ; sector count to r/w
00005262 894DFA         <1>     mov     [CMD_BLOCK+2], ecx ; LBA disk sector address
                        <1>
                        <1>     ;;;
                        <1>     ; 23/06/2024
00005265 C605[2C770100]FF <1>   mov     byte [LBAMode], 0FFh
                        <1>     ;
0000526C A0[72660000]   <1>     mov     al, [hf_m_s]
00005271 C0E004         <1>     shl     al, 4
                        <1>     ;add    al, 40h
00005274 0C40           <1>     or      al, 40h
                        <1>     ; al = 40h (for master), 50h (for slave)
00005276 A2[72660000]   <1>     mov     [hf_m_s], al
                        <1>     ;;;
                        <1>
0000527B 80FC1C         <1>     cmp     ah, 1Ch
0000527E 7509           <1>     jne     short su7
                        <1>     ; AH = 1Ch ; TRDOS 386 v2 - INT 33h - LBA WRITE
                        <1>     ; esi = sector buffer
                        <1>     ; 48 bit LBA write
00005280 C645FE34       <1>     mov     byte [CMD_BLOCK+6], 34h ; WRITE SECTOR(S) EXT
00005284 E9DD000000     <1>     jmp     CMD_WX
                        <1> su7:
                        <1>     ; AH = 1Bh ; TRDOS 386 v2 - INT 33h - LBA READ
00005289 89F7           <1>     mov     edi, esi ; sector buffer
                        <1>     ; 48 bit LBA read
0000528B C645FE24       <1>     mov     byte [CMD_BLOCK+6], 24h ; READ SECTOR(S) EXT
0000528F EB79           <1>     jmp     CMD_RX
                        <1>
                        <1> %endif
                        <1>
                        <1> ; ----------------------------------------------------
                        <1>
                        <1> ; 09/08/2022
                        <1> ; 07/08/2022
                        <1> ; 17/07/2022
                        <1> ; 16/07/2022 - TRDOS 386 v2.0.5
                        <1> ; 10/07/2022 - Retro UNIX 386 v1.1 (Kernel v0.2.1.5)
                        <1>
                        <1> ;----------------------------------------
                        <1> ;   RESET THE DISK SYSTEM  (AH=00H) :
                        <1> ;----------------------------------------
                        <1>
                        <1> ; 18-1-2015 : one controller reset (not other one)
                        <1>
                        <1> DISK_RESET:
00005291 FA             <1>     cli
00005292 E4A1           <1>     in      al, INTB01              ; GET THE MASK REGISTER
                        <1>     ;jmp    $+2
                        <1>     IODELAY
  78 00005294 EB00      <2>  jmp short $+2
  79 00005296 EB00      <2>  jmp short $+2
                        <1>     ;and    al, 0BFh                ; ENABLE FIXED DISK INTERRUPT
00005298 243F           <1>     and     al, 3Fh                 ; 22/12/2014 (IRQ 14 & IRQ 15)
0000529A E6A1           <1>     out     INTB01, al
0000529C FB             <1>     sti                             ; START INTERRUPTS
                        <1>     ; 14/02/2015
                        <1>     ;mov    di, dx
                        <1>     ; 24/12/2021
0000529D 89D7           <1>     mov     edi, edx
                        <1>     ; 04/01/2015
                        <1>     ;xor    di,di
                        <1> drst0:
0000529F B004           <1>     mov     al, 04h  ; bit 2 - SRST
                        <1>     ;mov    dx, HF_REG_PORT
000052A1 668B15[70660000] <1>   mov     dx, [HF_REG_PORT]
000052A8 EE             <1>     out     dx, al                  ; RESET
                        <1> ;   mov     cx, 10                  ; DELAY COUNT
                        <1> ;DRD:       dec     cx
                        <1> ;   jnz     short DRD               ; WAIT 4.8 MICRO-SEC
                        <1>     ;mov    cx, 2                   ; wait for 30 micro seconds
                        <1>         ;mov       ecx, 2 ; 21/02/2015
                        <1>     ; 10/07/2022
000052A9 29C9           <1>     sub     ecx, ecx
000052AB B102           <1>     mov     cl, 2
000052AD E86FD1FFFF     <1>     call    WAITF                     ; (Award Bios 1999 - WAIT_REFRESH,
                        <1>                                       ; 40 micro seconds)
000052B2 A0[19770100]   <1>     mov     al, [CONTROL_BYTE]
000052B7 240F           <1>     and     al, 0Fh                 ; SET HEAD OPTION
000052B9 EE             <1>     out     dx, al                  ; TURN RESET OFF
000052BA E85B040000     <1>     call    NOT_BUSY
000052BF 7514           <1>     jnz     short DRERR             ; TIME OUT ON RESET
000052C1 668B15[6E660000] <1>   mov     dx, [HF_PORT]
```

```
5302 000052C8 FEC2                <1>     inc     dl  ; HF_PORT+1
5303                              <1>     ; 02/01/2015 - Award BIOS 1999 - AHDSK.ASM
5304                              <1>         ;mov       cl, 10
5305                              <1>         ;mov       ecx, 10 ; 21/02/2015
5306                              <1>     ; 17/07/2022
5307                              <1>     ;xor    ecx, ecx
5308 000052CA B10A                <1>     mov     cl, 10
5309                              <1> drst1:
5310 000052CC EC                  <1>     in      al, dx                  ; GET RESET STATUS
5311 000052CD 3C01                <1>     cmp     al, 1
5312                              <1>     ; 04/01/2015
5313 000052CF 740C                <1>     jz      short drst2
5314                              <1>     ;jnz    short DRERR             ; BAD RESET STATUS
5315                              <1>             ; Drive/Head Register - bit 4
5316                              <1>     ;loop   drst1
5317                              <1>     ; 17/07/2022
5318 000052D1 FEC9                <1>     dec     cl
5319 000052D3 75F7                <1>     jnz     short drst1
5320                              <1> DRERR:
5321 000052D5 C605[17770100]05    <1>     mov     byte [DISK_STATUS1], BAD_RESET ; CARD FAILED
5322 000052DC C3                  <1>     retn
5323                              <1> drst2:
5324                              <1>     ; 14/02/2015
5325                              <1>     ;mov    dx, di
5326                              <1>     ; 07/08/2022
5327 000052DD 89FA                <1>     mov     edx, edi
5328                              <1> ;drst3:
5329                              <1> ;   ; 05/01/2015
5330                              <1> ;   shl     di, 1
5331                              <1> ;   ; 04/01/2015
5332                              <1> ;   mov     ax, [di+hd_cports]
5333                              <1> ;   cmp     ax, [HF_REG_PORT]
5334                              <1> ;   je      short drst4
5335                              <1> ;   mov     [HF_REG_PORT], ax
5336                              <1> ;   ; 03/01/2015
5337                              <1> ;   mov     ax, [di+hd_ports]
5338                              <1> ;       mov     [HF_PORT], ax
5339                              <1> ;   ; 05/01/2014
5340                              <1> ;   shr     di, 1
5341                              <1> ;   ; 04/01/2015
5342                              <1> ;   jmp     short drst0    ; reset other controller
5343                              <1> ;drst4:
5344                              <1> ;   ; 05/01/2015
5345                              <1> ;   shr     di, 1
5346                              <1> ;   mov     al, [di+hd_dregs]
5347                              <1> ;   and     al, 10h ; bit 4 only
5348                              <1> ;   shr     al, 4 ; bit 4  -> bit 0
5349                              <1> ;   mov     [hf_m_s], al ; (0 = master, 1 = slave)
5350                              <1>     ;
5351                              <1> ; 09/08/2022
5352                              <1> ; (('INIT_DRV' prodedure sets [CMD_BLOCKS+5] value))
5353                              <1> ;
5354                              <1> ;   mov     al, [hf_m_s] ; 18/01/2015
5355                              <1> ;   test    al, 1
5356                              <1> ;   ;jnz    short drst6
5357                              <1> ;       jnz     short drst4
5358                              <1> ;   and     byte [CMD_BLOCK+5], 0EFh ; SET TO DRIVE 0
5359                              <1> ;drst5:
5360                              <1> drst3:
5361 000052DF E8F9010000          <1>     call    INIT_DRV                ; SET MAX HEADS
5362                              <1>     ;mov    dx, di
5363 000052E4 E8C1020000          <1>     call    HDISK_RECAL             ; RECAL TO RESET SEEK SPEED
5364                              <1>     ; 04/01/2014
5365                              <1> ;   inc     di
5366                              <1> ;   mov     dx, di
5367                              <1> ;   cmp     dl, [HF_NUM]
5368                              <1> ;   jb      short drst3
5369                              <1> ;DRE:
5370 000052E9 C605[17770100]00    <1>     mov     byte [DISK_STATUS1], 0        ; IGNORE ANY SET UP ERRORS
5371 000052F0 C3                  <1>     retn
5372                              <1> ;drst6:
5373                              <1> drst4:              ; Drive/Head Register - bit 4
5374                              <1> ;   or      byte [CMD_BLOCK+5], 010h ; SET TO DRIVE 1
5375                              <1> ;       ;jmp    short drst5
5376                              <1> ;       jmp     short drst3
5377                              <1>
5378                              <1> ;----------------------------------------
5379                              <1> ;   DISK STATUS ROUTINE  (AH = 01H) :
5380                              <1> ;----------------------------------------
5381                              <1>
5382                              <1> RETURN_STATUS:
5383 000052F1 A0[17770100]        <1>     mov     al, [DISK_STATUS1]    ; OBTAIN PREVIOUS STATUS
5384 000052F6 C605[17770100]00    <1>     mov     byte [DISK_STATUS1], 0; RESET STATUS
5385 000052FD C3                  <1>     retn
5386                              <1>
5387                              <1>     ; 16/07/2022 - TRDOS 386 v2.0.5
5388                              <1>
5389                              <1> ;----------------------------------------
5390                              <1> ;   READ LONG           (AH = 0AH) :
5391                              <1> ;----------------------------------------
5392                              <1>
5393                              <1> RD_LONG:
5394                              <1>     ;mov    @CMD_BLOCK+6, READ_CMD OR ECC_MODE
5395 000052FE C645FE22            <1>     mov     byte [CMD_BLOCK+6], READ_CMD + ECC_MODE
5396 00005302 EB04                <1>     jmp     short COMMANDI
5397                              <1>
5398                              <1>     ; 16/07/2022 - TRDOS 386 v2.0.5
5399                              <1>
5400                              <1> ;----------------------------------------
5401                              <1> ;   DISK READ ROUTINE    (AH = 02H) :
5402                              <1> ;----------------------------------------
5403                              <1>
5404                              <1> DISK_READ:
5405 00005304 C645FE20            <1>     mov     byte [CMD_BLOCK+6], READ_CMD
5406                              <1>         ;jmp       COMMANDI
5407                              <1>
5408                              <1>     ; 16/07/2022 - TRDOS 386 v2.0.5
5409                              <1>
5410                              <1> ;----------------------------------------
5411                              <1> ; COMMANDI                          :
5412                              <1> ;   REPEATEDLY INPUTS DATA TILL   :
5413                              <1> ;   NSECTOR RETURNS ZERO          :
5414                              <1> ;----------------------------------------
5415                              <1> COMMANDI:
5416                              <1>     ; 16/07/2022
5417                              <1>     ;       (check 64K boundary is not needed)
5418                              <1>     ;call   CHECK_DMA               ; CHECK 64K BOUNDARY ERROR
5419                              <1>     ;jc     short CMD_ABORT
5420                              <1>
5421                              <1>     ;mov    di, bx
5422 00005308 89DF                <1>     mov     edi, ebx ; 21/02/2015
5423                              <1> CMD_RX:
5424                              <1>     ; 20/06/2024 (48 bit LBA r/w modification)
5425 0000530A E821030000          <1>     call    COMMAND                 ; OUTPUT COMMAND
```

```
5426 0000530F 7548              <1>     jnz     short CMD_ABORT
5427                            <1> ;CMD_RX: ; 01/12/2023 (48 bit LBA read)
5428                            <1> CMD_I1:
5429 00005311 E8D4030000        <1>     call    _WAIT                   ; WAIT FOR DATA REQUEST INTERRUPT
5430 00005316 7541              <1>     jnz     short TM_OUT            ; TIME OUT
5431                            <1> cmd_i1x:
5432                            <1>     ; 18/02/2016
5433                            <1>     ;;mov   cx, 256                 ; SECTOR SIZE IN WORDS
5434                            <1>     ;mov    ecx, 256 ; 21/02/2015
5435                            <1>     ; 16/07/2022
5436 00005318 29C9              <1>     sub     ecx, ecx
5437 0000531A FEC5              <1>     inc     ch  ; ecx = 256
5438                            <1>     ;mov    dh, HF_PORT
5439 0000531C 668B15[6E660000]  <1>     mov     dx, [HF_PORT]
5440 00005323 FA                <1>     cli
5441 00005324 FC                <1>     cld
5442 00005325 F3666D            <1>     rep     insw                    ; GET THE SECTOR
5443 00005328 FB                <1>     sti
5444                            <1>
5445 00005329 F645FE02          <1>     test    byte [CMD_BLOCK+6], ECC_MODE ; CHECK FOR NORMAL INPUT
5446 0000532D 7418              <1>     jz      short CMD_I3
5447 0000532F E80D040000        <1>     call    WAIT_DRQ                ; WAIT FOR DATA REQUEST
5448 00005334 7223              <1>     jc      short TM_OUT
5449                            <1>     ;mov    dx, HF_PORT
5450 00005336 668B15[6E660000]  <1>     mov     dx, [HF_PORT]
5451                            <1>     ;;mov   cx, 4                   ; GET ECC BYTES
5452                            <1>     ;mov    ecx, 4 ; mov cx, 4
5453                            <1>     ; 16/07/2022
5454 0000533D 31C9              <1>     xor     ecx, ecx
5455 0000533F B104              <1>     mov     cl, 4
5456                            <1> CMD_I2:
5457 00005341 EC                <1>     in      al, dx
5458 00005342 8807              <1>     mov     [edi], al ; 21/02/2015; GO SLOW FOR BOARD
5459 00005344 47                <1>     inc     edi
5460 00005345 E2FA              <1>     loop    CMD_I2
5461                            <1> CMD_I3:
5462                            <1>     ; wait for 400 ns
5463 00005347 80C207            <1>     add     dl, 7
5464 0000534A EC                <1>     in      al, dx
5465 0000534B EC                <1>     in      al, dx
5466 0000534C EC                <1>     in      al, dx
5467                            <1>     ;
5468 0000534D E8E4010000        <1>     call    CHECK_STATUS
5469 00005352 7505              <1>     jnz     short CMD_ABORT             ; ERROR RETURNED
5470 00005354 FE4DF9            <1>     dec     byte [CMD_BLOCK+1]    ; CHECK FOR MORE
5471                            <1>     ;jnz    short CMD_I1
5472 00005357 75BF              <1>     jnz     short cmd_i1x ; 18/02/2016
5473                            <1> CMD_ABORT:
5474                            <1> TM_OUT:
5475 00005359 C3                <1>     retn
5476                            <1>
5477                            <1>     ; 16/07/2022 - TRDOS 386 v2.0.5
5478                            <1>
5479                            <1> ;----------------------------------------
5480                            <1> ;   WRITE LONG          (AH = 0BH) :
5481                            <1> ;----------------------------------------
5482                            <1>
5483                            <1> WR_LONG:
5484                            <1>     ;mov    @CMD_BLOCK+6, WRITE_CMD OR ECC_MODE
5485 0000535A C645FE32          <1>     mov     byte [CMD_BLOCK+6], WRITE_CMD + ECC_MODE
5486 0000535E EB04              <1>     jmp     short COMMANDO
5487                            <1>
5488                            <1>     ; 16/07/2022 - TRDOS 386 v2.0.5
5489                            <1>
5490                            <1> ;----------------------------------------
5491                            <1> ;   DISK WRITE ROUTINE   (AH = 03H) :
5492                            <1> ;----------------------------------------
5493                            <1>
5494                            <1> DISK_WRITE:
5495 00005360 C645FE30          <1>     mov     byte [CMD_BLOCK+6], WRITE_CMD
5496                            <1>     ;jmp    COMMANDO
5497                            <1>
5498                            <1>     ; 16/07/2022 - TRDOS 386 v2.0.5
5499                            <1>
5500                            <1> ;----------------------------------------
5501                            <1> ; COMMANDO                          :
5502                            <1> ;   REPEATEDLY OUTPUTS DATA TILL  :
5503                            <1> ;   NSECTOR RETURNS ZERO          :
5504                            <1> ;----------------------------------------
5505                            <1> COMMANDO:
5506                            <1>     ; 16/07/2022
5507                            <1>     ;       (check 64K boundary is not needed)
5508                            <1>     ;call   CHECK_DMA               ; CHECK 64K BOUNDARY ERROR
5509                            <1>     ;jc     short CMD_ABORT
5510                            <1> CMD_OF:
5511 00005364 89DE              <1>     mov     esi, ebx ; 21/02/2015
5512                            <1> CMD_WX:
5513                            <1>     ; 20/06/2024 (48 bit LBA r/w modification)
5514 00005366 E8C5020000        <1>     call    COMMAND                 ; OUTPUT COMMAND
5515 0000536B 75EC              <1>     jnz     short CMD_ABORT
5516                            <1> ;CMD_WX:    ; 01/12/2023 (48 bit LBA write)
5517 0000536D E8CF030000        <1>     call    WAIT_DRQ                ; WAIT FOR DATA REQUEST
5518 00005372 72E5              <1>     jc      short TM_OUT            ; TOO LONG
5519                            <1> CMD_O1:
5520                            <1>     ; 16/07/2022
5521 00005374 668B15[6E660000]  <1>     mov     dx, [HF_PORT]
5522                            <1>
5523                            <1>     ;mov    ecx, 256 ; 21/02/2015
5524 0000537B 31C9              <1>     xor     ecx, ecx
5525 0000537D FEC5              <1>     inc     ch
5526                            <1>     ; ecx = 256
5527 0000537F FA                <1>     cli
5528 00005380 FC                <1>     cld
5529                            <1>     ;rep    outsw
5530                            <1>     ; 01/12/2023 - TRDOS 386 v2.0.7
5531                            <1> CMD_O1_L:
5532 00005381 666F              <1>     outsw
5533 00005383 EB00              <1>     jmp     $+2
5534 00005385 E2FA              <1>     loop    CMD_O1_L
5535                            <1>
5536 00005387 FB                <1>     sti
5537                            <1>
5538 00005388 F645FE02          <1>     test    byte [CMD_BLOCK+6], ECC_MODE ; CHECK FOR NORMAL OUTPUT
5539 0000538C 7418              <1>     jz      short CMD_O3
5540 0000538E E8AE030000        <1>     call    WAIT_DRQ                ; WAIT FOR DATA REQUEST
5541 00005393 72C4              <1>     jc      short TM_OUT
5542                            <1>     ;mov    dx, HF_PORT
5543 00005395 668B15[6E660000]  <1>     mov     dx, [HF_PORT]
5544                            <1>                                     ; OUTPUT THE ECC BYTES
5545                            <1>     ;mov    ecx, 4  ; mov cx, 4
5546                            <1>     ; 16/07/2022
5547 0000539C 29C9              <1>     sub     ecx, ecx
5548 0000539E B104              <1>     mov     cl, 4
5549                            <1> CMD_O2:
```

```
5550 000053A0 8A06                <1>     mov     al, [esi]
5551 000053A2 EE                  <1>     out     dx, al
5552 000053A3 46                  <1>     inc     esi
5553 000053A4 E2FA                <1>     loop    CMD_O2
5554                              <1> CMD_O3:
5555 000053A6 E83F030000          <1>     call    _WAIT                   ; WAIT FOR SECTOR COMPLETE INTERRUPT
5556 000053AB 75AC                <1>     jnz     short TM_OUT            ; ERROR RETURNED
5557 000053AD E884010000          <1>     call    CHECK_STATUS
5558 000053B2 75A5                <1>     jnz     short CMD_ABORT
5559 000053B4 F605[11770100]08    <1>     test    byte [HF_STATUS], ST_DRQ ; CHECK FOR MORE
5560 000053BB 75B7                <1>     jnz     short CMD_O1
5561                              <1>     ;mov    dx, HF_PORT+2           ; CHECK RESIDUAL SECTOR COUNT
5562 000053BD 668B15[6E660000]    <1>     mov     dx, [HF_PORT]
5563 000053C4 80C202              <1>     add     dl, 2
5564                              <1>     ;inc    dl
5565                              <1>     ;inc    dl
5566 000053C7 EC                  <1>     in      al, dx                  ;
5567 000053C8 A8FF                <1>     test    al, 0FFh                ;
5568 000053CA 7407                <1>     jz      short CMD_O4            ; COUNT = 0 OK
5569 000053CC C605[17770100]BB    <1>     mov     byte [DISK_STATUS1], UNDEF_ERR
5570                              <1>                                     ; OPERATION ABORTED - PARTIAL TRANSFER
5571                              <1> CMD_O4:
5572 000053D3 C3                  <1>     retn
5573                              <1>
5574                              <1>     ; 16/07/2022 - TRDOS 386 v2.0.5
5575                              <1>
5576                              <1> ;----------------------------------------
5577                              <1> ;   FORMATTING          (AH = 05H) :
5578                              <1> ;----------------------------------------
5579                              <1>
5580                              <1> FMT_TRK:                            ; FORMAT TRACK (AH = 005H)
5581 000053D4 C645FE50            <1>     mov     byte [CMD_BLOCK+6], FMTTRK_CMD
5582                              <1>     ;push   es
5583                              <1>     ;push   bx
5584 000053D8 53                  <1>     push    ebx
5585 000053D9 E8F1030000          <1>     call    GET_VEC                 ; GET DISK PARAMETERS ADDRESS
5586                              <1>     ;mov    al, [ES:BX+14]          ; GET SECTORS/TRACK
5587 000053DE 8A430E              <1>     mov     al, [ebx+14]
5588 000053E1 8845F9              <1>     mov     [CMD_BLOCK+1], al       ; SET SECTOR COUNT IN COMMAND
5589 000053E4 5B                  <1>     pop     ebx
5590                              <1>     ;pop    bx
5591                              <1>     ;pop    es
5592                              <1>     ;jmp    short CMD_OF            ; GO EXECUTE THE COMMAND
5593                              <1>     ; 01/12/2023
5594 000053E5 E97AFFFFFF          <1>     jmp     CMD_OF
5595                              <1>
5596                              <1> ;----------------------------------------
5597                              <1> ;   DISK VERIFY         (AH = 04H) :
5598                              <1> ;----------------------------------------
5599                              <1>
5600                              <1> DISK_VERF:
5601 000053EA C645FE40            <1>     mov     byte [CMD_BLOCK+6], VERIFY_CMD
5602 000053EE E83D020000          <1>     call    COMMAND
5603 000053F3 750C                <1>     jnz     short VERF_EXIT                 ; CONTROLLER STILL BUSY
5604 000053F5 E8F0020000          <1>     call    _WAIT                   ; (Original: CALL WAIT)
5605 000053FA 7505                <1>     jnz     short VERF_EXIT                 ; TIME OUT
5606 000053FC E835010000          <1>     call    CHECK_STATUS
5607                              <1> VERF_EXIT:
5608 00005401 C3                  <1>     retn
5609                              <1>
5610                              <1> ;----------------------------------------
5611                              <1> ;   READ DASD TYPE      (AH = 15H) :
5612                              <1> ;----------------------------------------
5613                              <1>
5614                              <1> RETURN_DRIVE_TYPE:
5615                              <1>     ; 10/08/2022
5616                              <1>     ; 13/07/2022
5617                              <1>     ; (Ref: Programmer's Guide to the AMIBIOS -Page 214-, 1992)
5618                              <1>     ;
5619                              <1>     ; INPUT:
5620                              <1>     ;       DL = Disk number (>= 80h)
5621                              <1>     ;   TRDOS 386 v2.0.5 Feature:
5622                              <1>     ;       If AL = 0FFh, return disk size in ECX
5623                              <1>     ;                 otherwise in CX:DX
5624                              <1>     ; OUTPUT:
5625                              <1>     ;       AH = 00h - No drive present
5626                              <1>     ;          = 03h - Hard disk drive
5627                              <1>     ;       CF = 0 - No error
5628                              <1>     ;          = 1 - Error
5629                              <1>     ;   TRDOS 386 v2.0.5 Feature:
5630                              <1>     ;       AL = 00h - LBA not ready !
5631                              <1>     ;          = 01h - LBA ready
5632                              <1>     ;               (28 bit or 48 bit LBA r/w depending
5633                              <1>     ;               on disk size in CX:DX)
5634                              <1>     ;       CX:DX = Number of 512 byte sectors
5635                              <1>     ;
5636                              <1>     ; (Note: High words of ECX and EDX will be zero at return)
5637                              <1>     ; ((If AL input is 0FFh, disk size will be in ECX only))
5638                              <1>
5639                              <1> READ_DASD_TYPE:
5640                              <1> READ_D_T:                           ; GET DRIVE PARAMETERS
5641                              <1>     ;push   ds                      ; SAVE REGISTERS
5642                              <1>
5643                              <1>     ;;push  es
5644                              <1>     ; 18/04/2021
5645                              <1>     ;push   ebx
5646                              <1>     ;;call  DDS                     ; ESTABLISH ADDRESSING
5647                              <1>     ;;push  cs
5648                              <1>     ;;pop   ds
5649                              <1>
5650                              <1>     ; 18/04/2021
5651                              <1>     ;mov    bx, KDATA
5652                              <1>     ;mov    ds, bx
5653                              <1>     ;;mov   es, bx
5654                              <1>     ;mov    byte [DISK_STATUS1], 0
5655                              <1>     ;mov    bl, [HF_NUM]            ; GET NUMBER OF DRIVES
5656                              <1>     ;and    dl, 7Fh                 ; GET DRIVE NUMBER
5657                              <1>     ;cmp    bl, dl
5658                              <1>     ;jbe    short RDT_NOT_PRESENT ; RETURN DRIVE NOT PRESENT
5659                              <1>
5660                              <1>     ;mov    ax, KDATA
5661                              <1>     ;mov    ds, ax
5662                              <1>
5663 00005402 C605[17770100]00    <1>     mov     byte [DISK_STATUS1], 0
5664 00005409 8A0D[18770100]      <1>     mov     cl, [HF_NUM]
5665 0000540F 80E27F              <1>     and     dl, 7Fh
5666 00005412 38D1                <1>     cmp     cl, dl
5667 00005414 7631                <1>     jbe     short RDT_NOT_PRESENT
5668                              <1>
5669                              <1>     ; 18/04/2021 - TRDOS 386 v2.0.4
5670                              <1>
5671                              <1>     ;call   GET_VEC                 ; GET DISK PARAMETER ADDRESS
5672                              <1>     ;
5673                              <1>     ;;mov   al, [ES:BX+2]           ; HEADS
```

```
                                <1>     ;mov    al, [ebx+2]  ; heads (logical)
                                <1>     ;;;mov  cl, [ES:BX+14]
                                <1>     ;;mov   cl, [ebx+14]
                                <1>     ;; 17/04/2021
                                <1>     ;mov    ah, [ebx+14] ; sectors per track (logical)
                                <1>     ;;imul  cl                   ; * NUMBER OF SECTORS
                                <1>     ;;mov   cx, [ES:BX]          ; MAX NUMBER OF CYLINDERS
                                <1>     ;mov    cx, [ebx]    ; cylinders (logical)
                                <1>     ;; 02/01/2015
                                <1>     ;; ** leave the last cylinder as reserved for diagnostics **
                                <1>     ;; (Also in Award BIOS - 1999, AHDSK.ASM, FUN15 -> sub ax, 1)
                                <1>     ;dec    cx                   ; LEAVE ONE FOR DIAGNOSTICS
                                <1>     ;;imul  cx                   ; NUMBER OF SECTORS
                                <1>     ;; 17/04/2021
                                <1>     ;mul    ah
                                <1>     ;; ax = spt*heads
                                <1>     ;mul    cx
                                <1>     ;; dx:ax = number of sectors
                                <1>     ;
                                <1>     ;mov    cx, dx               ; HIGH ORDER HALF
                                <1>     ;mov    dx, ax               ; LOW ORDER HALF
                                <1>
                                <1>     ; 18/04/2021
00005416 B102                   <1>     mov     cl, 2
00005418 00CA                   <1>     add     dl, cl ; hd0 = 2
                                <1>
                                <1>     ; 13/07/2022
0000541A 0FB6D2                 <1>     movzx   edx, dl
0000541D 8A9A[9E660000]         <1>     mov     bl, [edx+drv.status]
00005423 80E301                 <1>     and     bl, 1 ; LBA ready bit (bit 0)
                                <1>
00005426 D2E2                   <1>     shl     dl, cl ; * 4
                                <1>
                                <1>     ;mov    eax, [edx+drv.size]
                                <1>     ;mov    dx, ax
                                <1>     ;shr    eax, 16
                                <1>     ;mov    cx, ax
                                <1>
00005428 8B8A[82660000]         <1>     mov     ecx, [edx+drv.size]
                                <1>
                                <1>     ; 13/07/2022
                                <1>     ;cmp    al, 0FFh   ; return disk size in ecx ?
                                <1>     ;je     short RDT1 ; yes
0000542E FEC0                   <1>     inc     al  ; 0FFh -> 0
00005430 740A                   <1>     jz      short RDT1
                                <1>
00005432 6689CA                 <1>     mov     dx, cx
00005435 C1E910                 <1>     shr     ecx, 16
                                <1>
                                <1>     ; 13/07/2022
                                <1>     ; ebx = esp+20
                                <1>     ; ecx = esp+16
                                <1>     ; edx = esp+12
                                <1>     ; esi = esp+8
                                <1>     ; edi = esp+4
                                <1>
                                <1>     ; return disk size in user's registers
00005438 8954240C               <1>     mov     [esp+12], edx
                                <1>     ; cx:dx = disk size
                                <1> RDT1:
                                <1>     ;mov    [esp+16], ecx
                                <1>
                                <1>     ;;sub   al, al
0000543C 29C0                   <1>     sub     eax, eax
0000543E B403                   <1>     mov     ah, 03h              ; INDICATE FIXED DISK
00005440 88D8                   <1>     mov     al, bl
                                <1>     ; al = 1 -> LBA r/w ready/applicable
                                <1>     ;    = 0 -> LBA r/w not ready/applicable
                                <1>     ; cf = 0
                                <1> RDT2:
00005442 894C2410               <1>     mov     [esp+16], ecx
                                <1> ;RDT2:
                                <1>     ; 13/07/2022
                                <1>     ; 18/04/2021
                                <1>     ;pop    ebx                  ; RESTORE REGISTERS
                                <1>     ;;pop   es
                                <1>     ;pop    ds
                                <1>     ; (*) clc                    ; CLEAR CARRY
                                <1>     ;retf   2
                                <1>     ; (*) 29/05/2016
                                <1>     ; (*) retf 4
                                <1>     ;and    byte [esp+8], 0FEh ; clear carry bit of eflags register
                                <1>     ;iretd
                                <1>
                                <1>     ; 13/07/2022
                                <1>     ; [DISK_STATUS1] = 0
                                <1>     ; ah = 3
                                <1>     ; al = 0 or 1 (LBA ready)
                                <1>     ; cf = 0
                                <1>
00005446 C3                     <1>     retn
                                <1>
                                <1> RDT_NOT_PRESENT:
                                <1>     ;;sub   ax, ax               ; DRIVE NOT PRESENT RETURN
                                <1>     ;; 18/04/2021
                                <1>     ;sub    eax, eax
                                <1>     ;;mov   cx, ax               ; ZERO BLOCK COUNT
                                <1>     ;;mov   dx, ax
                                <1>     ;mov    ecx, eax
                                <1>     ;mov    edx, eax
                                <1>     ;jmp    short RDT2
                                <1>     ; 13/07/2022
00005447 29C9                   <1>     sub     ecx, ecx
00005449 88C1                   <1>     mov     cl, al ; if AL = 0FFh, disk size will be in ECX
                                <1>                    ; if not, disk size will be in CX:DX
0000544B 29C0                   <1>     sub     eax, eax
0000544D FEC1                   <1>     inc     cl ; 0FFh -> 0
0000544F 80F901                 <1>     cmp     cl, 1
00005452 72EE                   <1>     jb      short RDT2 ; ecx = 0
                                <1>     ; 10/08/2022
00005454 28C9                   <1>     sub     cl, cl
                                <1>     ; ecx = 0
00005456 8944240C               <1>     mov     [esp+12], eax ; edx = 0
0000545A F9                     <1>     stc
0000545B EBE5                   <1>     jmp     short RDT2 ; cf = 1, eax = 0
                                <1>
                                <1> ; 10/08/2022
                                <1> ; 07/08/2022
                                <1> ; 13/07/2022 - TRDOS 386 v2.0.5
                                <1> ; 28/05/2016
                                <1> ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>
                                <1> ;----------------------------------------
                                <1> ;   GET PARAMETERS      (AH = 08H) :
```

```
                            <1> ;----------------------------------------
                            <1>
                            <1> GET_PARM_N:
                            <1>     ; ebx = user's buffer address for parameters table
                            <1>     ; 10/08/2022
                            <1>     ; 13/07/2022
                            <1>     ; (if ebx = 0, HDPT will not be returned to user)
                            <1>     ;
                            <1> ;GET_PARM:                          ; GET DRIVE PARAMETERS
                            <1>     ;push  ds                       ; SAVE REGISTERS
                            <1>     ;push  es
                            <1>
                            <1>     ;push  ebx
0000545D 89DF               <1>     mov    edi, ebx ; 13/07/2022
                            <1>
                            <1>     ; 13/07/2022
                            <1>     ; ((IBM PC XT-286 ROM BIOS source code remainders))
                            <1>     ;;mov  ax, ABS0                 ; ESTABLISH ADDRESSING
                            <1>     ;;mov  ds, ax
                            <1>
                            <1>     ;;test dl, 1                    ; CHECK FOR DRIVE 1
                            <1>     ;;jz   short G0
                            <1>     ;;les  bx, @HF1_TBL_VEC
                            <1>     ;;jmp  short G1
                            <1> ;;G0:
                            <1>     ;les   bx, @HF_TBL_VEC
                            <1> ;;G1:
                            <1>     ;;call DDS                      ; ESTABLISH SEGMENT
                            <1>
                            <1>     ; 13/07/2022
                            <1>     ; 22/12/2014
                            <1>     ;;push cs
                            <1>     ;;pop  ds
                            <1>     ;mov   bx, KDATA
                            <1>     ;mov   ds, bx
                            <1>     ;mov   es, bx  ; 27/05/2016
                            <1>     ;
                            <1>     ; 18/04/2021
0000545F 29C9               <1>     sub    ecx, ecx
                            <1>     ;
00005461 80EA80             <1>     sub    dl, 80h
00005464 80FA04             <1>     cmp    dl, MAX_FILE             ; TEST WITHIN RANGE
00005467 7344               <1>     jae    short G2 ; 13/07/2022
                            <1>     ;
                            <1>     ; 21/02/2015
00005469 31DB               <1>     xor    ebx, ebx
                            <1>     ; 18/04/2021
                            <1>     ;sub   ecx, ecx
                            <1>     ; 22/12/2014
0000546B 88D3               <1>     mov    bl, dl
                            <1>     ;xor   bh, bh
0000546D C0E302             <1>     shl    bl, 2                    ; convert index to offset
                            <1>     ;add   bx, HF_TBL_VEC
00005470 81C3[1C770100]     <1>     add    ebx, HF_TBL_VEC
                            <1>     ;mov   ax, [bx+2]
                            <1>     ;mov   es, ax                   ; dpt segment
                            <1>     ;mov   bx, [bx]                 ; dpt offset
00005476 8B1B               <1>     mov    ebx, [ebx] ; 32 bit offset
                            <1>     ; 18/04/2021
00005478 29D2               <1>     sub    edx, edx
0000547A 8815[17770100]     <1>     mov    [DISK_STATUS1], dl ; 0
                            <1>
                            <1>     ;mov   byte [DISK_STATUS1], 0
                            <1>         ;mov       ax, [ES:BX]          ; MAX NUMBER OF CYLINDERS
00005480 668B03             <1>     mov    ax, [ebx]
                            <1>     ;;sub  ax, 2                    ; ADJUST FOR 0-N
00005483 6648               <1>     dec    ax                       ; max. cylinder number
00005485 88C5               <1>     mov    ch, al
00005487 66250003           <1>     and    ax, 0300h                ; HIGH TWO BITS OF CYLINDER
                            <1>     ;shr   ax, 1
                            <1>     ;shr   ax, 1
                            <1>     ; 13/07/2022
                            <1>     ;shr   ax, 2
                            <1>     ; 07/08/2022
0000548B C1E802             <1>     shr    eax, 2
                            <1>     ;or    al, [ES:BX+14]           ; SECTORS
0000548E 0A430E             <1>     or     al, [ebx+14]
00005491 88C1               <1>     mov    cl, al
                            <1>     ;mov   dh, [ES:BX+2]            ; HEADS
00005493 8A7302             <1>     mov    dh, [ebx+2]
00005496 FECE               <1>     dec    dh                       ; 0-N RANGE
00005498 8A15[18770100]     <1>     mov    dl, [HF_NUM]             ; DRIVE COUNT
                            <1>     ;;sub  ax, ax
                            <1>     ; 18/04/2021
                            <1>     ;sub   eax, eax
                            <1>
                            <1>     ; 27/12/2014
                            <1>     ;mov   di, bx                   ; HDPT offset
                            <1>
                            <1>     ; 13/07/2022
                            <1>     ; ebx = esp+20
                            <1>     ; ecx = esp+16
                            <1>     ; edx = esp+12
                            <1>     ; esi = esp+8
                            <1>     ; edi = esp+4
                            <1>
                            <1>     ; 13/07/2022
                            <1>     ; set return register contents/values
0000549E 894C2410           <1>     mov    [esp+16], ecx
000054A2 8954240C           <1>     mov    [esp+12], edx
                            <1>
                            <1>     ; is hard disk parameters table requested ?
000054A6 09FF               <1>     or     edi, edi ; (edi = [ebp+24] = ebx)
000054A8 751B               <1>     jnz    short G3 ; yes
                            <1>
000054AA 29C0               <1>     sub    eax, eax
                            <1>
                            <1>     ; [DISK_STATUS1] = 0
                            <1>     ; eax = 0
                            <1>     ; cf = 0
                            <1>
000054AC C3                 <1>     retn
                            <1>
                            <1> G2:
                            <1>     ;mov   ah, INIT_FAIL
                            <1>     ;mov   byte [DISK_STATUS1], ah ; (INIT_FAIL)
                            <1>     ;                               ; OPERATION FAILED
                            <1>     ;sub   al, al
                            <1>     ;sub   dx, dx
                            <1>     ;sub   cx, cx
                            <1>     ; 18/04/2021
000054AD 29C0               <1>     sub    eax, eax
000054AF B407               <1>     mov    ah, INIT_FAIL
000054B1 8825[17770100]     <1>     mov     [DISK_STATUS1], ah    ; OPERATION FAILED
```

```
                                <1>
                                <1>     ; 13/07/2022
                                <1>     ;sub    edx, edx
                                <1>     ;sub    ecx, ecx
                                <1>     ; ecx = 0
000054B7 894C240C               <1>     mov     [esp+12], ecx ; 0 ; edx (heads-1, drive count)
000054BB 894C2410               <1>     mov     [esp+16], ecx ; 0 ; ecx (cylinders-1, sectors)
000054BF 894C2414               <1>     mov     [esp+20], ecx ; 0 ; ebx (HDPT address)
                                <1>
000054C3 F9                     <1>     stc
000054C4 C3                     <1>     retn
                                <1>
                                <1>     ; 13/07/2022
                                <1>     ; 29/05/2016 (*)
                                <1>     ;;stc                           ; SET ERROR FLAG
                                <1>     ;;jmp   short G5
                                <1>     ;jmp    short _G6
                                <1>
                                <1> G3:
                                <1>     ; 27/05/2016
                                <1>     ; return fixed disk parameters table to user
                                <1>     ; in user's buffer, which is pointed by EBX
                                <1>
                                <1>     ;xchg   edi, [esp]              ; ebx (input)-> edi, edi -> [esp]
                                <1>     ; 13/07/2022
                                <1>     ;pop    edi
                                <1>     ; edi = user's buffer address
                                <1>     ;push   esi
000054C5 89DE                   <1>     mov     esi, ebx                ; hard disk parameter table (32 bytes)
                                <1>     ;mov    ebx, edi                ; ebx = user's buffer address
000054C7 51                     <1>     push    ecx
                                <1>     ;push   eax
                                <1>     ;mov    ecx, 32 ; 32 bytes
000054C8 30ED                   <1>     xor     ch, ch
000054CA B120                   <1>     mov     cl, 32
                                <1>     ; ecx = 32
000054CC E819B60000             <1>     call    transfer_to_user_buffer ; trdosk6.s (16/05/2016)
                                <1>     ;pop    eax
000054D1 59                     <1>     pop     ecx
                                <1>     ; 10/08/2022
                                <1>     ;pop    esi
                                <1>     ;pop    edi
000054D2 7306                   <1>     jnc     short G4
                                <1>     ; 29/05/2016 (*)
000054D4 B8FF000000             <1>     mov     eax, 0FFh ; unknown error !
                                <1>     ; [DISK_STATUS1] = 0
                                <1>     ; ah = 0, al = 0FFh
                                <1>     ; cf = 1
                                <1>
000054D9 C3                     <1>     retn
                                <1> ;_G6:
                                <1>     ;or     byte [esp+16], 1 ; set carry bit of eflags register
                                <1> ;G5:
                                <1>     ; 27/05/2016
                                <1>     ;pop    ebx                     ; RESTORE REGISTERS
                                <1>     ;pop    es
                                <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
                                <1>     ;iretd
                                <1>
                                <1> G4:
                                <1>     ; 13/07/2022
000054DA 31C0                   <1>     xor     eax, eax
                                <1>
                                <1>     ; [user_buffer] = [ebp+24] = HDPT
                                <1>     ; [DISK_STATUS1] = 0
                                <1>     ; eax = 0
                                <1>     ; cf = 0
                                <1>
000054DC C3                     <1>     retn
                                <1>
                                <1>     ; 16/07/2022 - TRDOS 386 v2.0.5
                                <1>
                                <1> ;----------------------------------------
                                <1> ;   INITIALIZE DRIVE     (AH = 09H) :
                                <1> ;----------------------------------------
                                <1>     ; 03/01/2015
                                <1>     ; According to ATA-ATAPI specification v2.0 to v5.0
                                <1>     ; logical sector per logical track
                                <1>     ; and logical heads - 1 would be set but
                                <1>     ; it is seen as it will be good
                                <1>     ; if physical parameters will be set here
                                <1>     ; because, number of heads <= 16.
                                <1>     ; (logical heads usually more than 16)
                                <1>     ; NOTE: ATA logical parameters (software C, H, S)
                                <1>     ;       == INT 13h physical parameters
                                <1>
                                <1> ;INIT_DRV:
                                <1> ;   mov     byte [CMD_BLOCK+6], SET_PARM_CMD
                                <1> ;   call    GET_VEC                 ; ES:BX -> PARAMETER BLOCK
                                <1> ;   mov     al, [es:bx+2]           ; GET NUMBER OF HEADS
                                <1> ;   dec     al                      ; CONVERT TO 0-INDEX
                                <1> ;   mov     ah, [CMD_BLOCK+5]       ; GET SDH REGISTER
                                <1> ;   and     ah, 0F0h                ; CHANGE HEAD NUMBER
                                <1> ;   or      ah, al                  ; TO MAX HEAD
                                <1> ;   mov     [CMD_BLOCK+5], ah
                                <1> ;   mov     al, [es:bx+14]          ; MAX SECTOR NUMBER
                                <1> ;   mov     [CMD_BLOCK+1], al
                                <1> ;   sub     ax, ax
                                <1> ;   mov     [CMD_BLOCK+3], al       ; ZERO FLAGS
                                <1> ;   call    COMMAND                 ; TELL CONTROLLER
                                <1> ;   jnz     short INIT_EXIT                 ; CONTROLLER BUSY ERROR
                                <1> ;   call    NOT_BUSY                ; WAIT FOR IT TO BE DONE
                                <1> ;   jnz     short INIT_EXIT                 ; TIME OUT
                                <1> ;   call    CHECK_STATUS
                                <1> ;INIT_EXIT:
                                <1> ;   retn
                                <1>
                                <1> ; 16/07/2022 - TRDOS 386 v2.0.5
                                <1>
                                <1> ; 04/01/2015
                                <1> ; 02/01/2015 - Derived from from AWARD BIOS 1999
                                <1> ;                           AHDSK.ASM - INIT_DRIVE
                                <1> INIT_DRV:
                                <1>     ;xor    ah,ah
000054DD 31C0                   <1>     xor     eax, eax ; 21/02/2015
000054DF B00B                   <1>     mov     al, 11 ; Physical heads from translated HDPT
000054E1 3825[2C770100]         <1>         cmp     [LBAMode], ah   ; 0
000054E7 7702                   <1>     ja      short idrv0
000054E9 B002                   <1>     mov     al, 2  ; Physical heads from standard HDPT
                                <1> idrv0:
```

```
                                <1>     ; DL = drive number (0 based)
000054EB E8DF020000             <1>     call    GET_VEC
                                <1>     ;push   bx
000054F0 53                     <1>     push    ebx ; 21/02/2015
                                <1>     ;add    bx, ax
000054F1 01C3                   <1>     add     ebx, eax
                                <1>     ;; 05/01/2015
000054F3 8A25[72660000]         <1>     mov     ah, [hf_m_s] ; drive number (0= master, 1= slave)
                                <1>     ;;and   ah, 1
000054F9 C0E404                 <1>     shl     ah, 4
000054FC 80CCA0                 <1>     or      ah, 0A0h  ; Drive/Head register - 10100000b (A0h)
                                <1>     ;mov    al, [es:bx]
000054FF 8A03                   <1>     mov     al, [ebx] ; 21/02/2015
00005501 FEC8                   <1>     dec     al        ; last head number
                                <1>     ;and    al, 0Fh
00005503 08E0                   <1>     or      al, ah   ; lower 4 bits for head number
                                <1>     ;
00005505 C645FE91               <1>     mov     byte [CMD_BLOCK+6], SET_PARM_CMD
00005509 8845FD                 <1>     mov     [CMD_BLOCK+5], al
                                <1>     ;pop    bx
0000550C 5B                     <1>     pop     ebx
0000550D 29C0                   <1>     sub     eax, eax ; 21/02/2015
0000550F B004                   <1>     mov     al, 4   ; Physical sec per track from translated HDPT
00005511 803D[2C770100]00       <1>     cmp     byte [LBAMode], 0
00005518 7702                   <1>     ja      short idrv1
0000551A B00E                   <1>     mov     al, 14  ; Physical sec per track from standard HDPT
                                <1> idrv1:
                                <1>     ;xor    ah, ah
                                <1>     ;add    bx, ax
0000551C 01C3                   <1>     add     ebx, eax ; 21/02/2015
                                <1>     ;mov    al, [es:bx]
                                <1>                      ; sector number
0000551E 8A03                   <1>     mov     al, [ebx]
00005520 8845F9                 <1>     mov     [CMD_BLOCK+1], al
00005523 28C0                   <1>     sub     al, al
00005525 8845FB                 <1>     mov     [CMD_BLOCK+3], al ; ZERO FLAGS
00005528 E803010000             <1>     call    COMMAND           ; TELL CONTROLLER
0000552D 751E                   <1>     jnz     short INIT_EXIT           ; CONTROLLER BUSY ERROR
0000552F E8E6010000             <1>     call    NOT_BUSY          ; WAIT FOR IT TO BE DONE
00005534 7517                   <1>     jnz     short INIT_EXIT           ; TIME OUT
                                <1>     ; 16/07/2022
                                <1>     ;call   CHECK_STATUS
                                <1>     ;jmp    short CHECK_STATUS
                                <1> ;INIT_EXIT:
                                <1>     ;retn
                                <1>
                                <1>     ; 16/07/2022 - TRDOS 386 v2.0.5
                                <1>     ; 10/07/2022 - Retro UNIX 386 v1.1 (Kernel v0.2.1.5)
                                <1>
                                <1> ;----------------------------------------
                                <1> ;   CHECK FIXED DISK STATUS       :
                                <1> ;----------------------------------------
                                <1> CHECK_STATUS:
00005536 E831020000             <1>     call    CHECK_ST                  ; CHECK THE STATUS BYTE
                                <1>     ;jnz    short CHECK_S1            ; AN ERROR WAS FOUND
                                <1>     ; 10/07/2022
0000553B 7510                   <1>     jnz     short CHECK_S2
0000553D A801                   <1>     test    al, ST_ERROR              ; WERE THERE ANY OTHER ERRORS
0000553F 7405                   <1>     jz      short CHECK_S1            ; NO ERROR REPORTED
00005541 E866020000             <1>     call    CHECK_ER                  ; ERROR REPORTED
                                <1> CHECK_S1:
00005546 803D[17770100]00       <1>     cmp     byte [DISK_STATUS1], 0        ; SET STATUS FOR CALLER
                                <1> CHECK_S2:
                                <1> INIT_EXIT:  ; 10/07/2022
0000554D C3                     <1>     retn
                                <1>
                                <1>     ; 16/07/2022 - TRDOS 386 v2.0.5
                                <1>
                                <1> ;----------------------------------------
                                <1> ;   SEEK                (AH = 0CH) :
                                <1> ;----------------------------------------
                                <1>
                                <1> DISK_SEEK:
0000554E C645FE70               <1>         mov byte [CMD_BLOCK+6], SEEK_CMD
00005552 E8D9000000             <1>     call    COMMAND
00005557 751C                   <1>     jnz     short DS_EXIT             ; CONTROLLER BUSY ERROR
00005559 E88C010000             <1>     call    _WAIT
0000555E 7515                   <1>         jnz short DS_EXIT             ; TIME OUT ON SEEK
00005560 E8D1FFFFFF             <1>     call    CHECK_STATUS
00005565 803D[17770100]40       <1>         cmp byte [DISK_STATUS1], BAD_SEEK
0000556C 7507                   <1>     jne     short DS_EXIT
0000556E C605[17770100]00       <1>         mov byte [DISK_STATUS1], 0
                                <1> DS_EXIT:
00005575 C3                     <1>     retn
                                <1>
                                <1> ;----------------------------------------
                                <1> ;   TEST DISK READY      (AH = 10H) :
                                <1> ;----------------------------------------
                                <1>
                                <1> TST_RDY:                              ; WAIT FOR CONTROLLER
00005576 E89F010000             <1>     call    NOT_BUSY
0000557B 752C                   <1>     jnz     short TR_EX
                                <1>     ;;;
                                <1>     ; 23/06/2024
0000557D 803D[2C770100]FF       <1>     cmp     byte [LBAMode], 0FFh
00005584 7507                   <1>     jne     short tst_28bit_rdy
00005586 A0[72660000]           <1>     mov     al, [hf_m_s]
0000558B EB03                   <1>     jmp     short tst_48bit_rdy
                                <1> tst_28bit_rdy:
                                <1>     ;;;
0000558D 8A45FD                 <1>     mov     al, [CMD_BLOCK+5]     ; SELECT DRIVE
                                <1> tst_48bit_rdy:
00005590 668B15[6E660000]       <1>     mov     dx, [HF_PORT]
00005597 80C206                 <1>     add     dl, 6
0000559A EE                     <1>     out     dx, al
0000559B E8CC010000             <1>     call    CHECK_ST                  ; CHECK STATUS ONLY
000055A0 7507                   <1>     jnz     short TR_EX
000055A2 C605[17770100]00       <1>     mov     byte [DISK_STATUS1], 0        ; WIPE OUT DATA CORRECTED ERROR
                                <1> TR_EX:
000055A9 C3                     <1>     retn
                                <1>
                                <1> ;----------------------------------------
                                <1> ;   RECALIBRATE         (AH = 11H) :
                                <1> ;----------------------------------------
                                <1>
                                <1> HDISK_RECAL:
000055AA C645FE10               <1>         mov byte [CMD_BLOCK+6], RECAL_CMD ; 10h, 16
000055AE E87D000000             <1>     call    COMMAND                   ; START THE OPERATION
000055B3 7523                   <1>     jnz     short RECAL_EXIT          ; ERROR
000055B5 E830010000             <1>     call    _WAIT                     ; WAIT FOR COMPLETION
000055BA 7407                   <1>     jz      short RECAL_X             ; TIME OUT ONE OK ?
000055BC E829010000             <1>     call    _WAIT                     ; WAIT FOR COMPLETION LONGER
000055C1 7515                   <1>     jnz     short RECAL_EXIT          ; TIME OUT TWO TIMES IS ERROR
                                <1> RECAL_X:
```

```
6170 000055C3 E86EFFFFFF          <1>     call    CHECK_STATUS
6171 000055C8 803D[17770100]40    <1>     cmp     byte [DISK_STATUS1], BAD_SEEK ; SEEK NOT COMPLETE
6172 000055CF 7507                <1>     jne     short RECAL_EXIT      ; IS OK
6173 000055D1 C605[17770100]00    <1>     mov     byte [DISK_STATUS1], 0
6174                              <1> RECAL_EXIT:
6175 000055D8 803D[17770100]00    <1>         cmp byte [DISK_STATUS1], 0
6176 000055DF C3                  <1>     retn
6177                              <1>
6178                              <1> ;----------------------------------------
6179                              <1> ;       CONTROLLER DIAGNOSTIC (AH = 14H) :
6180                              <1> ;----------------------------------------
6181                              <1>
6182                              <1> CTLR_DIAGNOSTIC:
6183                              <1>     ; 07/08/2022 - TRDOS 386 v2.0.5
6184 000055E0 FA                  <1>     cli                           ; DISABLE INTERRUPTS WHILE CHANGING MASK
6185 000055E1 E4A1                <1>     in      al, INTB01            ; TURN ON SECOND INTERRUPT CHIP
6186                              <1>     ;and    al, 0BFH
6187 000055E3 243F                <1>     and     al, 3Fh               ; enable IRQ 14 & IRQ 15
6188                              <1>     ;jmp    $+2
6189                              <1>     IODELAY
  78 000055E5 EB00                <2>  jmp short $+2
  79 000055E7 EB00                <2>  jmp short $+2
6190 000055E9 E6A1                <1>     out     INTB01, al
6191                              <1>     IODELAY
  78 000055EB EB00                <2>  jmp short $+2
  79 000055ED EB00                <2>  jmp short $+2
6192 000055EF E421                <1>     in      al, INTA01            ; LET INTERRUPTS PASS THRU TO
6193 000055F1 24FB                <1>     and     al, 0FBh              ;  SECOND CHIP
6194                              <1>     ;jmp    $+2
6195                              <1>     IODELAY
  78 000055F3 EB00                <2>  jmp short $+2
  79 000055F5 EB00                <2>  jmp short $+2
6196 000055F7 E621                <1>     out     INTA01, al
6197 000055F9 FB                  <1>     sti
6198 000055FA E81B010000          <1>     call    NOT_BUSY              ; WAIT FOR CARD
6199 000055FF 7526                <1>     jnz     short CD_ERR          ; BAD CARD
6200                              <1>     ;mov    dx, PORT+7
6201 00005601 668B15[6E660000]    <1>     mov     dx, [HF_PORT]
6202 00005608 80C207              <1>     add     dl, 7
6203 0000560B B090                <1>     mov     al, DIAG_CMD          ; START DIAGNOSE
6204 0000560D EE                  <1>     out     dx, al
6205 0000560E E807010000          <1>     call    NOT_BUSY              ; WAIT FOR IT TO COMPLETE
6206 00005613 B480                <1>     mov     ah, TIME_OUT
6207 00005615 7512                <1>     jnz     short CD_EXIT         ; TIME OUT ON DIAGNOSTIC
6208                              <1>     ;mov    dx, HF_PORT+1         ; GET ERROR REGISTER
6209 00005617 668B15[6E660000]    <1>     mov     dx, [HF_PORT]
6210 0000561E FEC2                <1>     inc     dl
6211 00005620 EC                  <1>     in      al, dx
6212                              <1>     ; 07/08/2022
6213                              <1>     ;mov    [HF_ERROR], al        ; SAVE IT
6214 00005621 B400                <1>     mov     ah, 0
6215 00005623 3C01                <1>     cmp     al, 1                 ; CHECK FOR ALL OK
6216 00005625 7402                <1>     je      short CD_EXIT
6217                              <1> CD_ERR:
6218 00005627 B420                <1>     mov     ah, BAD_CNTLR
6219                              <1> CD_EXIT:
6220 00005629 8825[17770100]      <1>     mov     [DISK_STATUS1], ah
6221 0000562F C3                  <1>     retn
6222                              <1>
6223                              <1>     ; 20/06/2024 - TRDOS 386 v2.0.8
6224                              <1>     ; 16/07/2022 - TRDOS 386 v2.0.5
6225                              <1>     ; 10/07/2022 - Retro UNIX 386 v1.1 (Kernel v0.2.1.5)
6226                              <1>
6227                              <1> ;--------------------------------------------------------
6228                              <1> ; COMMAND                                              :
6229                              <1> ;   THIS ROUTINE OUTPUTS THE COMMAND BLOCK                :
6230                              <1> ; OUTPUT                                               :
6231                              <1> ;   BL = STATUS                                        :
6232                              <1> ;   BH = ERROR REGISTER                                :
6233                              <1> ;--------------------------------------------------------
6234                              <1>
6235                              <1> COMMAND:
6236                              <1>     ;push   ebx ; 10/07/2022      ; WAIT FOR SEEK COMPLETE AND READY
6237                              <1>     ;;mov   ecx, DELAY_2          ; SET INITIAL DELAY BEFORE TEST
6238                              <1> COMMAND1:
6239                              <1>     ;;push  ecx                   ; SAVE LOOP COUNT
6240 00005630 E841FFFFFF          <1>     call    TST_RDY               ; CHECK DRIVE READY
6241                              <1>     ;;pop   ecx
6242                              <1>     ;pop    ebx ; 10/07/2022
6243 00005635 7418                <1>     jz      short COMMAND2        ; DRIVE IS READY
6244 00005637 803D[17770100]80    <1>         cmp byte [DISK_STATUS1], TIME_OUT ; TST_RDY TIMED OUT--GIVE UP
6245                              <1>     ;jz     short CMD_TIMEOUT
6246                              <1>     ;;loop  COMMAND1              ; KEEP TRYING FOR A WHILE
6247                              <1>     ;jmp    short COMMAND4        ; ITS NOT GOING TO GET READY
6248 0000563E 7507                <1>     jne     short COMMAND4
6249                              <1> CMD_TIMEOUT:
6250 00005640 C605[17770100]20    <1>     mov     byte [DISK_STATUS1], BAD_CNTLR
6251                              <1> COMMAND4:
6252                              <1>     ;;pop   ebx ; 10/07/2022
6253 00005647 803D[17770100]00    <1>         cmp byte [DISK_STATUS1], 0; SET CONDITION CODE FOR CALLER
6254 0000564E C3                  <1>     retn
6255                              <1> COMMAND2:
6256                              <1>     ;;pop   ebx ; 10/07/2022
6257                              <1>     ;push   edi ; 10/07/2022
6258 0000564F C605[12770100]00    <1>     mov     byte [HF_INT_FLAG], 0 ; RESET INTERRUPT FLAG
6259 00005656 FA                  <1>     cli                           ; INHIBIT INTERRUPTS WHILE CHANGING MASK
6260 00005657 E4A1                <1>     in      al, INTB01            ; TURN ON SECOND INTERRUPT CHIP
6261                              <1>     ;and    al, 0BFh
6262 00005659 243F                <1>     and     al, 3Fh               ; Enable IRQ 14 & 15
6263                              <1>     ;jmp    $+2
6264                              <1>     IODELAY
  78 0000565B EB00                <2>  jmp short $+2
  79 0000565D EB00                <2>  jmp short $+2
6265 0000565F E6A1                <1>     out     INTB01, al
6266 00005661 E421                <1>     in      al, INTA01            ; LET INTERRUPTS PASS THRU TO
6267 00005663 24FB                <1>     and     al, 0FBh              ; SECOND CHIP
6268                              <1>     ;jmp    $+2
6269                              <1>     IODELAY
  78 00005665 EB00                <2>  jmp short $+2
  79 00005667 EB00                <2>  jmp short $+2
6270 00005669 E621                <1>     out     INTA01, al
6271 0000566B FB                  <1>     sti
6272                              <1>     ;xor    edi, edi              ; INDEX THE COMMAND TABLE
6273                              <1>     ; 10/07/2022
6274 0000566C 31C9                <1>     xor     ecx, ecx
6275                              <1>
6276                              <1>     ;mov    dx, HF_PORT+1         ; DISK ADDRESS
6277 0000566E 668B15[6E660000]    <1>     mov     dx, [HF_PORT]
6278 00005675 FEC2                <1>     inc     dl
6279                              <1>
6280                              <1>     ; 20/06/2024
6281                              <1>     ;mov    al, [CMD_BLOCK+6]
6282                              <1>     ;cmp    al, 24h ; READ SECTOR(S) EXT
6283                              <1>     ;je     short COMMAND5
```

```
                            <1>     ;cmp    al, 34h ; WRITE SECTOR(S) EXT
                            <1>     ;je     short COMMAND5
                            <1>     ; 23/06/2024
00005677 803D[2C770100]FF   <1>     cmp     byte [LBAMode], 0FFh
0000567E 7428               <1>     je      short COMMAND5 ; 48 bit LBA read/write
                            <1>
                            <1>     ; 20/06/2024
                            <1>     ; dx = hd base port + 1 ; 1F1h or 171h
                            <1>
00005680 F605[19770100]C0   <1>     test    byte [CONTROL_BYTE], 0C0h ; CHECK FOR RETRY SUPPRESSION
00005687 740E               <1>     jz      short COMMAND3
                            <1>     ; 20/06/2024
                            <1>     ;mov    al, [CMD_BLOCK+6]      ; YES-GET OPERATION CODE
00005689 24F0               <1>     and     al, 0F0h               ; GET RID OF MODIFIERS
0000568B 3C20               <1>     cmp     al, 20h                ; 20H-40H IS READ, WRITE, VERIFY
0000568D 7208               <1>     jb      short COMMAND3
0000568F 3C40               <1>     cmp     al, 40h
00005691 7704               <1>     ja      short COMMAND3
00005693 804DFE01           <1>     or      byte [CMD_BLOCK+6], NO_RETRIES
                            <1>                                    ; VALID OPERATION FOR RETRY SUPPRESS
                            <1> COMMAND3:
                            <1>     ;mov    al, [CMD_BLOCK+edi]    ; GET THE COMMAND STRING BYTE
                            <1>     ; 10/07/2022
00005697 8A440DF8           <1>     mov     al, [CMD_BLOCK+ecx]
0000569B EE                 <1>     out     dx, al                 ; GIVE IT TO CONTROLLER
                            <1>     IODELAY
  78 0000569C EB00          <2>  jmp short $+2
  79 0000569E EB00          <2>  jmp short $+2
                            <1>     ;inc    edi                    ; NEXT BYTE IN COMMAND BLOCK
                            <1>     ; 10/07/2022
000056A0 41                 <1>     inc     ecx
                            <1>     ;inc    dx                     ; NEXT DISK ADAPTER REGISTER
000056A1 42                 <1>     inc     edx    ; 10/07/2022
                            <1>     ;cmp    di, 7 ; 01/01/2015     ; ALL DONE?
                            <1>     ;jne    short COMMAND3         ; NO--GO DO NEXT ONE
000056A2 80F907             <1>     cmp     cl, 7 ; 10/07/2022
000056A5 72F0               <1>     jb      short COMMAND3
                            <1>     ;pop    edi ; 10/07/2022
000056A7 C3                 <1>     retn                           ; ZERO FLAG IS SET
                            <1>
                            <1>     ; 20/06/2024
                            <1> COMMAND5:
                            <1>     ; 48 bit LBA r/w
000056A8 8A45F8             <1>     mov     al, [CMD_BLOCK] ; 0
000056AB EE                 <1>     out     dx, al  ; hd base port + 1
                            <1>     ;
000056AC 80C205             <1>     add     dl, 5   ; hd base port + 6 ; 1F6h or 176h
000056AF A0[72660000]       <1>     mov     al, [hf_m_s]
                            <1>     ; 23/06/2024
                            <1>     ;shl    al, 4
                            <1>     ;;add   al, 40h
                            <1>     ;or     al, 40h
                            <1>     ;; al = 40h (for master), 50h (for slave)
                            <1>     ;
000056B4 EE                 <1>     out     dx, al
000056B5 80EA04             <1>     sub     dl, 4   ; hd base port + 2 ; 1F2h or 172h
000056B8 30C0               <1>     xor     al, al  ; 0
000056BA EE                 <1>     out     dx, al  ; sector count hb (bits 8 to 15) = 0
000056BB 42                 <1>     inc     edx     ; hd base port + 3
000056BC 8B45FA             <1>     mov     eax, [CMD_BLOCK+2]
000056BF C1C008             <1>     rol     eax, 8
000056C2 EE                 <1>     out     dx, al  ; LBA byte 4 (bits 24 to 31)
000056C3 42                 <1>     inc     edx     ; hd base port + 4
000056C4 30C0               <1>     xor     al, al
000056C6 EE                 <1>     out     dx, al  ; LBA byte 5 (bits 32 to 39) = 0
000056C7 42                 <1>     inc     edx     ; hd base port + 5
000056C8 28C0               <1>     sub     al, al
000056CA EE                 <1>     out     dx, al  ; LBA byte 6 (bits 40 to 47) = 0
000056CB 8A45F9             <1>     mov     al, [CMD_BLOCK+1] ; sector count
000056CE 80EA03             <1>     sub     dl, 3   ; hd base port + 2
000056D1 EE                 <1>     out     dx, al  ; sector count lb (bits 0 to 7)
000056D2 C1E808             <1>     shr     eax, 8
000056D5 42                 <1>     inc     edx     ; hd base port + 3 ; 1F3h or 173h
000056D6 EE                 <1>     out     dx, al  ; LBA byte 1 (bits 0 to 7)
000056D7 C1E808             <1>     shr     eax, 8
000056DA 42                 <1>     inc     edx     ; hd base port + 4
000056DB EE                 <1>     out     dx, al  ; LBA byte 2 (bits 8 to 15)
000056DC C1E808             <1>     shr     eax, 8
000056DF 42                 <1>     inc     edx     ; hd base port + 5
000056E0 EE                 <1>     out     dx, al  ; LBA byte 3 (bits 16 to 23
000056E1 42                 <1>     inc     edx     ; hd base port + 6
000056E2 8A45FE             <1>     mov     al, [CMD_BLOCK+6] ; 48 bit read or write command
                            <1>     ; al = 24h or 34h
000056E5 42                 <1>     inc     edx     ; hd base port + 7 ; 1F7h or 177h
000056E6 EE                 <1>     out     dx, al
000056E7 31C0               <1>     xor     eax, eax ; 0
                            <1>     ; zf = 1
000056E9 C3                 <1>     retn
                            <1>
                            <1> ;CMD_TIMEOUT:
                            <1> ;   mov     byte [DISK_STATUS1], BAD_CNTLR
                            <1> ;COMMAND4:
                            <1> ;   pop     ebx
                            <1> ;   cmp     byte [DISK_STATUS1], 0       ; SET CONDITION CODE FOR CALLER
                            <1> ;   retn
                            <1>
                            <1>     ; 16/07/2022 - TRDOS 386 v2.0.5
                            <1>     ; 10/07/2022 - Retro UNIX 386 v1.1 (Kernel v0.2.1.5)
                            <1>
                            <1> ;----------------------------------------
                            <1> ;   WAIT FOR INTERRUPT           :
                            <1> ;----------------------------------------
                            <1> ;WAIT:
                            <1> _WAIT:
000056EA 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     short 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
000056EB 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
000056F0 F605[12770100]C0   <1>     test    byte [HF_INT_FLAG], 0C0h
                            <1>     ;loopz  WT1
```

```
6406 000056F7 7512                <1>     jnz     short WT3               ; INTERRUPT--LETS GO
6407                              <1>     ;dec    bl
6408                              <1>     ;jnz    short WT1               ; KEEP TRYING FOR A WHILE
6409                              <1>
6410                              <1> WT1_hi:
6411 000056F9 E461                <1>     in      al, SYS1 ; 61h (PORT_B)       ; wait for lo to hi
6412 000056FB A810                <1>     test    al, 10h                 ; transition on memory
6413 000056FD 75FA                <1>     jnz     short WT1_hi            ; refresh.
6414                              <1> WT1_lo:
6415 000056FF E461                <1>     in      al, SYS1                ; 061h (PORT_B)
6416 00005701 A810                <1>     test    al, 10h
6417 00005703 74FA                <1>     jz      short WT1_lo
6418 00005705 E2E9                <1>     loop    WT1
6419                              <1>     ;;or    bl, bl
6420                              <1>     ;;jz    short WT2
6421                              <1>     ;;dec   bl
6422                              <1>     ;;jmp   short WT1
6423                              <1>     ;dec    bl
6424                              <1>     ;jnz    short WT1
6425                              <1> WT2:
6426                              <1>     ; 10/07/2022
6427                              <1>     ;mov    byte [DISK_STATUS1], TIME_OUT ; REPORT TIME OUT ERROR
6428 00005707 B080                <1>     mov     al, TIME_OUT
6429 00005709 EB07                <1>     jmp     short WT4
6430                              <1> WT3:
6431                              <1>     ;mov    byte [DISK_STATUS1], 0
6432                              <1>     ;mov    byte [HF_INT_FLAG], 0
6433 0000570B 28C0                <1>     sub     al, al ; 0
6434 0000570D A2[12770100]        <1>     mov     byte [HF_INT_FLAG], al
6435                              <1> WT4:
6436                              <1> NB2:
6437 00005712 A2[17770100]        <1>     mov     byte [DISK_STATUS1], al
6438                              <1>
6439                              <1>     ;cmp    byte [DISK_STATUS1], 0        ; SET CONDITION CODE FOR CALLER
6440 00005717 20C0                <1>     and     al, al
6441                              <1>     ; zf = 0 -> time out, zf = 1 -> ok
6442 00005719 C3                  <1>     retn
6443                              <1>
6444                              <1>     ; 16/07/2022 - TRDOS 386 v2.0.5
6445                              <1>     ; 10/07/2022 - Retro UNIX 386 v1.1 (Kernel v0.2.1.5)
6446                              <1>
6447                              <1> ;----------------------------------------
6448                              <1> ;   WAIT FOR CONTROLLER NOT BUSY  :
6449                              <1> ;----------------------------------------
6450                              <1> NOT_BUSY:
6451 0000571A FB                  <1>     sti                             ; MAKE SURE INTERRUPTS ARE ON
6452                              <1>     ;push   ebx
6453                              <1>     ;sub    cx, cx                  ; SET INITIAL DELAY BEFORE TEST
6454 0000571B 668B15[6E660000]    <1>     mov     dx, [HF_PORT]
6455 00005722 80C207              <1>     add     dl, 7                   ; Status port (HF_PORT+7)
6456                              <1>     ;mov    bl, DELAY_1
6457                              <1>                                     ; wait for 10 seconds
6458                              <1>     ;mov    cx, WAIT_HDU_INT_LO     ; 1615h
6459                              <1>     ;;mov   bl, WAIT_HDU_INT_HI     ;   05h
6460                              <1>     ;mov    bl, WAIT_HDU_INT_HI + 1
6461 00005725 B915160500          <1>     mov     ecx, WAIT_HDU_INT_LH    ; 21/02/2015
6462                              <1>     ;
6463                              <1>     ;;mov   byte [wait_count], 0    ; Reset wait counter
6464                              <1> NB1:
6465 0000572A EC                  <1>     in      al, dx                  ; CHECK STATUS
6466                              <1>     ;test   al, ST_BUSY
6467 0000572B 2480                <1>     and     al, ST_BUSY
6468                              <1>     ;loopnz NB1
6469 0000572D 74E3                <1>     jz      short NB2 ; al = 0      ; NOT BUSY--LETS GO
6470                              <1>     ;dec    bl
6471                              <1>     ;jnz    short NB1               ; KEEP TRYING FOR A WHILE
6472                              <1>
6473                              <1> NB1_hi:
6474 0000572F E461                <1>     in      al, SYS1                ; wait for hi to lo
6475 00005731 A810                <1>     test    al, 010h                ; transition on memory
6476 00005733 75FA                <1>     jnz     short NB1_hi            ; refresh.
6477                              <1> NB1_lo:
6478 00005735 E461                <1>     in      al, SYS1
6479 00005737 A810                <1>     test    al, 010h
6480 00005739 74FA                <1>     jz      short NB1_lo
6481 0000573B E2ED                <1>     loop    NB1
6482                              <1>     ;dec    bl
6483                              <1>     ;jnz    short NB1
6484                              <1>     ;
6485                              <1>     ;;cmp   byte [wait_count], 182  ; 10 seconds (182 timer ticks)
6486                              <1>     ;;jb    short NB1
6487                              <1>     ;
6488                              <1>     ;mov    byte [DISK_STATUS1], TIME_OUT ; REPORT TIME OUT ERROR
6489                              <1>     ;jmp    short NB3
6490 0000573D B080                <1>     mov     al, TIME_OUT
6491                              <1> ;NB2:
6492 0000573F EBD1                <1>     jmp     short NB2 ; 10/07/2022
6493                              <1>
6494                              <1> ;   ;mov    byte [DISK_STATUS1], 0
6495                              <1> ;;NB3:
6496                              <1> ;   ;pop    ebx
6497                              <1> ;   mov     [DISK_STATUS1], al    ;;; will be set after return
6498                              <1> ;   ;cmp    byte [DISK_STATUS1], 0        ; SET CONDITION CODE FOR CALLER
6499                              <1> ;   or      al, al                  ; (zf = 0 --> timeout)
6500                              <1> ;   retn
6501                              <1>
6502                              <1> ;----------------------------------------
6503                              <1> ;   WAIT FOR DATA REQUEST         :
6504                              <1> ;----------------------------------------
6505                              <1> WAIT_DRQ:
6506                              <1>     ;mov    cx, DELAY_3
6507                              <1>     ;mov    dx, HF_PORT+7
6508 00005741 668B15[6E660000]    <1>     mov     dx, [HF_PORT]
6509 00005748 80C207              <1>     add     dl, 7
6510                              <1>     ;;mov   bl, WAIT_HDU_DRQ_HI     ; 0
6511                              <1>     ;mov    cx, WAIT_HDU_DRQ_LO     ; 1000 (30 milli seconds)
6512                              <1>                                     ; (but it is written as 2000
6513                              <1>                                     ; micro seconds in ATORGS.ASM file
6514                              <1>                                     ; of Award Bios - 1999, D1A0622)
6515 0000574B B9E8030000          <1>     mov     ecx, WAIT_HDU_DRQ_LH ; 21/02/2015
6516                              <1> WQ_1:
6517 00005750 EC                  <1>     in      al, dx                  ; GET STATUS
6518 00005751 A808                <1>     test    al, ST_DRQ              ; WAIT FOR DRQ
6519 00005753 7516                <1>     jnz     short WQ_OK
6520                              <1>     ;loop   WQ_1                    ; KEEP TRYING FOR A SHORT WHILE
6521                              <1> WQ_hi:
6522 00005755 E461                <1>     in      al, SYS1                ; wait for hi to lo
6523 00005757 A810                <1>     test    al, 010h                ; transition on memory
6524 00005759 75FA                <1>     jnz     short WQ_hi             ; refresh.
6525                              <1> WQ_lo:
6526 0000575B E461                <1>     in      al, SYS1
6527 0000575D A810                <1>     test    al, 010h
6528 0000575F 74FA                <1>     jz      short WQ_lo
6529 00005761 E2ED                <1>     loop    WQ_1
```

```
6530                                    <1>
6531 00005763 C605[17770100]80          <1>     mov     byte [DISK_STATUS1], TIME_OUT ; ERROR
6532 0000576A F9                        <1>     stc
6533                                    <1> WQ_OK:
6534 0000576B C3                        <1>     retn
6535                                    <1> ;WQ_OK:
6536                                    <1>     ;clc
6537                                    <1>     ;ret
6538                                    <1>
6539                                    <1>     ; 16/07/2022 - TRDOS 386 v2.0.5
6540                                    <1>
6541                                    <1> ;----------------------------------------
6542                                    <1> ;   CHECK FIXED DISK STATUS BYTE  :
6543                                    <1> ;----------------------------------------
6544                                    <1> CHECK_ST:
6545                                    <1>     ;mov    dx, HF_PORT+7           ; GET THE STATUS
6546 0000576C 668B15[6E660000]          <1>     mov     dx, [HF_PORT]
6547 00005773 80C207                    <1>     add     dl, 7
6548                                    <1>
6549                                    <1>     ; 17/02/2016
6550                                    <1>     ;(http://wiki.osdev.org/ATA_PIO_Mode)
6551                                    <1>     ;"delay 400ns to allow drive to set new values of BSY and DRQ"
6552 00005776 EC                        <1>     in      al, dx
6553                                    <1>     ;in     al, dx ; 100ns
6554                                    <1>     ;in     al, dx ; 100ns
6555                                    <1>     ;in     al, dx ; 100ns
6556                                    <1>     NEWIODELAY ; 18/02/2016 (AWARD BIOS - 1999, 'CKST' in AHSDK.ASM)
  83 00005777 E6EB                      <2>  out 0EBh, al
6557                                    <1>     ;
6558                                    <1>
6559                                    <1>     ; 16/07/2022
6560 00005779 A2[11770100]              <1>     mov     [HF_STATUS], al
6561                                    <1>     ;mov    ah, 0
6562 0000577E 28E4                      <1>     sub     ah, ah ; 0
6563 00005780 A880                      <1>     test    al, ST_BUSY             ; IF STILL BUSY
6564 00005782 751A                      <1>     jnz     short CKST_EXIT                 ; REPORT OK
6565 00005784 B4CC                      <1>     mov     ah, WRITE_FAULT
6566 00005786 A820                      <1>     test    al, ST_WRT_FLT          ; CHECK FOR WRITE FAULT
6567 00005788 7514                      <1>     jnz     short CKST_EXIT
6568 0000578A B4AA                      <1>     mov     ah, NOT_RDY
6569 0000578C A840                      <1>     test    al, ST_READY            ; CHECK FOR NOT READY
6570 0000578E 740E                      <1>     jz      short CKST_EXIT
6571 00005790 B440                      <1>     mov     ah, BAD_SEEK
6572 00005792 A810                      <1>     test    al, ST_SEEK_COMPL       ; CHECK FOR SEEK NOT COMPLETE
6573 00005794 7408                      <1>     jz      short CKST_EXIT
6574 00005796 B411                      <1>     mov     ah, DATA_CORRECTED
6575 00005798 A804                      <1>     test    al, ST_CORRCTD          ; CHECK FOR CORRECTED ECC
6576 0000579A 7502                      <1>     jnz     short CKST_EXIT
6577                                    <1>     ;mov    ah, 0
6578 0000579C 30E4                      <1>     xor     ah, ah ; 0
6579                                    <1> CKST_EXIT:
6580 0000579E 8825[17770100]            <1>     mov     [DISK_STATUS1], ah      ; SET ERROR FLAG
6581 000057A4 80FC11                    <1>     cmp     ah, DATA_CORRECTED      ; KEEP GOING WITH DATA CORRECTED
6582 000057A7 7402                      <1>     je      short CKST_EX1
6583                                    <1>     ;cmp    ah, 0
6584 000057A9 20E4                      <1>     and     ah, ah
6585                                    <1> CKST_EX1:
6586 000057AB C3                        <1>     retn
6587                                    <1>
6588                                    <1>     ; 16/07/2022 - TRDOS 386 v2.0.5
6589                                    <1>
6590                                    <1> ;----------------------------------------
6591                                    <1> ;   CHECK FIXED DISK ERROR REGISTER :
6592                                    <1> ;----------------------------------------
6593                                    <1> CHECK_ER:
6594                                    <1>     ;mov    dx, HF_PORT+1           ; GET THE ERROR REGISTER
6595 000057AC 668B15[6E660000]          <1>     mov     dx, [HF_PORT]           ;
6596 000057B3 FEC2                      <1>     inc     dl
6597 000057B5 EC                        <1>     in      al, dx
6598                                    <1>     ; 16/07/2022
6599                                    <1>     ;mov    [HF_ERROR], al
6600                                    <1>     ;push   ebx  ; 21/02/2015
6601 000057B6 29C9                      <1>     sub     ecx, ecx
6602                                    <1>     ;mov    ecx, 8                  ; TEST ALL 8 BITS
6603 000057B8 B108                      <1>     mov     cl, 8
6604                                    <1> CK1:
6605 000057BA D0E0                      <1>     shl     al, 1                   ; MOVE NEXT ERROR BIT TO CARRY
6606 000057BC 7202                      <1>     jc      short CK2               ; FOUND THE ERROR
6607 000057BE E2FA                      <1>     loop    CK1                     ; KEEP TRYING
6608                                    <1> CK2:
6609                                    <1>     ;mov    ebx, ERR_TBL            ; COMPUTE ADDRESS OF
6610                                    <1>     ;add    ebx, ecx                ; ERROR CODE
6611 000057C0 81C1[64660000]            <1>     add     ecx, ERR_TBL ; 16/07/2022
6612                                    <1>
6613                                    <1>     ;;;mov  ah, byte [cs:bx]        ; GET ERROR CODE
6614                                    <1>     ;;mov   ah, [bx]
6615                                    <1>     ;mov    ah, [ebx] ; 21/02/2015
6616 000057C6 8A21                      <1>     mov     ah, [ecx]
6617                                    <1> CKEX:
6618 000057C8 8825[17770100]            <1>     mov     [DISK_STATUS1], ah      ; SAVE ERROR CODE
6619                                    <1>     ; 16/07/2022
6620                                    <1>     ;pop    ebx
6621                                    <1>     ;;cmp   ah, 0
6622                                    <1>     ;and    ah, ah
6623 000057CE C3                        <1>     retn
6624                                    <1>
6625                                    <1> ;--------------------------------------------------------
6626                                    <1> ; CHECK_DMA                                            :
6627                                    <1> ;  -CHECK ES:BX AND # SECTORS TO MAKE SURE THAT IT WILL :
6628                                    <1> ;   FIT WITHOUT SEGMENT OVERFLOW.                      :
6629                                    <1> ;  -ES:BX HAS BEEN REVISED TO THE FORMAT SSSS:000X     :
6630                                    <1> ;  -OK IF # SECTORS < 80H (7FH IF LONG READ OR WRITE)  :
6631                                    <1> ;  -OK IF # SECTORS = 80H (7FH) AND BX <= 00H (04H)    :
6632                                    <1> ;  -ERROR OTHERWISE                                    :
6633                                    <1> ;--------------------------------------------------------
6634                                    <1>
6635                                    <1>     ; 16/07/2022 - TRDOS 386 v2.0.5
6636                                    <1>     ; (not needed for hard disks and 32 bit OS)
6637                                    <1>
6638                                    <1> ;CHECK_DMA:
6639                                    <1> ;   ; 11/04/2021 (32 bit push/pop)
6640                                    <1> ;   push    eax                     ; SAVE REGISTERS
6641                                    <1> ;   mov     ax, 8000h               ; AH = MAX # SECTORS
6642                                    <1> ;                                   ; AL = MAX OFFSET
6643                                    <1> ;   test    byte [CMD_BLOCK+6], ECC_MODE
6644                                    <1> ;   jz      short CKD1
6645                                    <1> ;   mov     ah, 7F04h               ; ECC IS 4 MORE BYTES
6646                                    <1> ;CKD1:
6647                                    <1> ;   cmp     ah, [CMD_BLOCK+1]       ; NUMBER OF SECTORS
6648                                    <1> ;   ja      short CKDOK             ; IT WILL FIT
6649                                    <1> ;   jb      short CKDERR            ; TOO MANY
6650                                    <1> ;   cmp     al, bl                  ; CHECK OFFSET ON MAX SECTORS
6651                                    <1> ;   ;jb     short CKDERR            ; ERROR
6652                                    <1> ;   jnb     short CKDOK ; 16/07/2022
```

```
6653                            <1> ;;CKDOK:
6654                            <1> ;   ;clc                            ; CLEAR CARRY
6655                            <1> ;   ;pop    eax
6656                            <1> ;   ;retn                           ; NORMAL RETURN
6657                            <1> ;CKDERR:
6658                            <1> ;   ;stc                            ; INDICATE ERROR
6659                            <1> ;   mov     byte [DISK_STATUS1], DMA_BOUNDARY
6660                            <1> ;CKDOK:
6661                            <1> ;   pop     eax
6662                            <1> ;   retn
6663                            <1>
6664                            <1> ;----------------------------------------
6665                            <1> ;   SET UP EBX-> DISK PARMS              :
6666                            <1> ;----------------------------------------
6667                            <1>
6668                            <1> ; INPUT -> DL = 0 based drive number
6669                            <1> ; OUTPUT -> EBX = disk parameter table address
6670                            <1>
6671                            <1> GET_VEC:
6672                            <1>     ;sub    ax, ax                  ; GET DISK PARAMETER ADDRESS
6673                            <1>     ;mov    es, ax
6674                            <1>     ;test   dl, 1
6675                            <1>     ;jz     short GV_0
6676                            <1> ;   les     bx, [HF1_TBL_VEC]       ; ES:BX -> DRIVE PARAMETERS
6677                            <1> ;   jmp     short GV_EXIT
6678                            <1> ;GV_0:
6679                            <1> ;   les     bx,[HF_TBL_VEC]                 ; ES:BX -> DRIVE PARAMETERS
6680                            <1> ;
6681 000057CF 31DB             <1>     xor     ebx, ebx
6682 000057D1 88D3             <1>     mov     bl, dl
6683                            <1>     ;02/01/2015
6684                            <1>     ;xor    bh, bh
6685                            <1>     ;shl    bl, 1                   ; port address offset
6686                            <1>     ;mov    ax, [bx+hd_ports]       ; Base port address (1F0h, 170h)
6687                            <1>     ;shl    bl, 1                   ; dpt pointer offset
6688 000057D3 C0E302           <1>     shl     bl, 2
6689                            <1>     ;add    bx, HF_TBL_VEC          ; Disk parameter table pointer
6690 000057D6 81C3[1C770100]   <1>     add     ebx, HF_TBL_VEC ; 21/02/2015
6691                            <1>     ;push   word [bx+2]             ; dpt segment
6692                            <1>     ;pop    es
6693                            <1>     ;mov    bx, [bx]                ; dpt offset
6694 000057DC 8B1B             <1>     mov     ebx, [ebx]
6695                            <1> ;GV_EXIT:
6696 000057DE C3               <1>     retn
6697                            <1>
6698                            <1> hdc1_int: ; 21/02/2015
6699                            <1> ;--- HARDWARE INT 76H -- ( IRQ LEVEL 14 ) ------------------------
6700                            <1> ;                                                                 :
6701                            <1> ;   FIXED DISK INTERRUPT ROUTINE                                  :
6702                            <1> ;                                                                 :
6703                            <1> ;-----------------------------------------------------------------
6704                            <1>
6705                            <1> ; 22/12/2014
6706                            <1> ; IBM PC-XT Model 286 System BIOS Source Code - DISK.ASM (HD_INT)
6707                            <1> ;    '11/15/85'
6708                            <1> ; AWARD BIOS 1999 (D1A0622)
6709                            <1> ;   Source Code - ATORGS.ASM (INT_HDISK, INT_HDISK1)
6710                            <1>
6711                            <1> ;int_76h:
6712                            <1> HD_INT:
6713                            <1>     ; 11/04/2021 (32 bit push/pop)
6714 000057DF 50               <1>     push    eax
6715 000057E0 1E               <1>     push    ds
6716                            <1>     ;call   DDS
6717                            <1>     ; 21/02/2015 (32 bit, 386 pm modification)
6718 000057E1 66B81000         <1>     mov     ax, KDATA
6719 000057E5 8ED8             <1>     mov     ds, ax
6720                            <1>     ;
6721                            <1>     ;;mov   @HF_INT_FLAG,0FFH       ; ALL DONE
6722                            <1>         ;mov       byte [CS:HF_INT_FLAG], 0FFh
6723 000057E7 C605[12770100]FF <1>     mov     byte [HF_INT_FLAG], 0FFh
6724                            <1>     ;
6725 000057EE 52               <1>     push    edx
6726 000057EF 66BAF701         <1>     mov     dx, HDC1_BASEPORT+7     ; Status Register (1F7h)
6727                            <1>                                     ; Clear Controller
6728                            <1> Clear_IRQ1415:                              ; (Award BIOS - 1999)
6729 000057F3 EC               <1>     in      al, dx                  ;
6730 000057F4 5A               <1>     pop     edx
6731                            <1>     NEWIODELAY
  83 000057F5 E6EB             <2>  out 0EBh, al
6732                            <1>     ;
6733 000057F7 B020             <1>     mov     al, EOI                 ; NON-SPECIFIC END OF INTERRUPT
6734 000057F9 E6A0             <1>     out     INTB00, al              ; FOR CONTROLLER #2
6735                            <1>     ;jmp    $+2                     ; WAIT
6736                            <1>     NEWIODELAY
  83 000057FB E6EB             <2>  out 0EBh, al
6737 000057FD E620             <1>     out     INTA00, al              ; FOR CONTROLLER #1
6738 000057FF 1F               <1>     pop     ds
6739                            <1>     ;sti                            ; RE-ENABLE INTERRUPTS
6740                            <1>     ;mov    ax, 9100h               ; DEVICE POST
6741                            <1>     ;int    15h                     ;  INTERRUPT
6742                            <1> irq15_iret: ; 25/02/2015
6743 00005800 58               <1>     pop     eax
6744 00005801 CF               <1>     iretd                           ; RETURN FROM INTERRUPT
6745                            <1>
6746                            <1> hdc2_int: ; 21/02/2015
6747                            <1> ;--- HARDWARE INT 77H -- ( IRQ LEVEL 15 ) ------------------------
6748                            <1> ;                                                                 :
6749                            <1> ;   FIXED DISK INTERRUPT ROUTINE                                  :
6750                            <1> ;                                                                 :
6751                            <1> ;-----------------------------------------------------------------
6752                            <1>
6753                            <1> ;int_77h:
6754                            <1> HD1_INT:
6755                            <1>     ; 11/04/2021 (32 bit push/pop)
6756 00005802 50               <1>     push    eax
6757                            <1>     ; Check if that is a spurious IRQ (from slave PIC)
6758                            <1>     ; 25/02/2015 (source: http://wiki.osdev.org/8259_PIC)
6759 00005803 B00B             <1>     mov     al, 0Bh  ; In-Service Register
6760 00005805 E6A0             <1>     out     0A0h, al
6761 00005807 EB00             <1>         jmp short $+2
6762 00005809 EB00             <1>     jmp short $+2
6763 0000580B E4A0             <1>     in      al, 0A0h
6764 0000580D 2480             <1>     and     al, 80h ; bit 7 (is it real IRQ 15 or fake?)
6765 0000580F 74EF             <1>     jz      short irq15_iret ; Fake (spurious) IRQ, do not send EOI)
6766                            <1>     ;
6767 00005811 1E               <1>     push    ds
6768                            <1>     ;call   DDS
6769                            <1>     ; 21/02/2015 (32 bit, 386 pm modification)
6770 00005812 66B81000         <1>     mov     ax, KDATA
6771 00005816 8ED8             <1>     mov     ds, ax
6772                            <1>     ;
6773                            <1>     ;;mov   @HF_INT_FLAG,0FFH       ; ALL DONE
6774                            <1>         ;or byte [CS:HF_INT_FLAG],0C0h
```

```
6775 00005818 800D[12770100]C0    <1>     or      byte [HF_INT_FLAG], 0C0h
6776                              <1>     ;
6777 0000581F 52                  <1>     push    edx
6778 00005820 66BA7701            <1>     mov     dx, HDC2_BASEPORT+7   ; Status Register (177h)
6779                              <1>                                   ; Clear Controller (Award BIOS 1999)
6780 00005824 EBCD                <1>     jmp     short Clear_IRQ1415
6781                              <1>
6782                              <1> ;%include 'diskdata.inc' ; 11/03/2015
6783                              <1> ;%include 'diskbss.inc' ; 11/03/2015
6784                              <1>
6785                              <1> ;////////////////////////////////////////////////////////////////
6786                              <1> ;; END OF DISK I/O SYTEM ///
3228                                  %include 'memory.s'  ; 09/03/2015
   1                              <1> ; ****************************************************************************
   2                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.8 - memory.s
   3                              <1> ; ----------------------------------------------------------------------------
   4                              <1> ; Last Update: 04/06/2024  (Previous: 02/12/2023 - Kernel v2.0.7)
   5                              <1> ; ----------------------------------------------------------------------------
   6                              <1> ; Beginning: 24/01/2016
   7                              <1> ; ----------------------------------------------------------------------------
   8                              <1> ; Assembler: NASM version 2.15 (trdos386.s)
   9                              <1> ; ----------------------------------------------------------------------------
  10                              <1> ; Turkish Rational DOS
  11                              <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
  12                              <1> ;
  13                              <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
  14                              <1> ; memory.inc (18/10/2015)
  15                              <1> ; ****************************************************************************
  16                              <1>
  17                              <1> ; MEMORY.ASM - Retro UNIX 386 v1 MEMORY MANAGEMENT FUNCTIONS (PROCEDURES)
  18                              <1> ; Retro UNIX 386 v1 Kernel (unix386.s, v0.2.0.14) - MEMORY.INC
  19                              <1> ; Last Modification: 18/10/2015
  20                              <1>
  21                              <1> ; ///////// MEMORY MANAGEMENT FUNCTIONS (PROCEDURES) ///////////////
  22                              <1>
  23                              <1> ;;04/11/2014 (unix386.s)
  24                              <1> ;PDE_A_PRESENT     equ     1               ; Present flag for PDE
  25                              <1> ;PDE_A_WRITE       equ     2               ; Writable (write permission) flag
  26                              <1> ;PDE_A_USER equ     4               ; User (non-system/kernel) page flag
  27                              <1> ;;
  28                              <1> ;PTE_A_PRESENT     equ     1               ; Present flag for PTE (bit 0)
  29                              <1> ;PTE_A_WRITE       equ     2               ; Writable (write permission) flag (bit 1)
  30                              <1> ;PTE_A_USER equ     4               ; User (non-system/kernel) page flag (bit 2)
  31                              <1> ;PTE_A_ACCESS   equ       32              ; Accessed flag (bit 5) ; 09/03/2015
  32                              <1>
  33                              <1> ; 27/04/2015
  34                              <1> ; 09/03/2015
  35                              <1> PAGE_SIZE   equ 4096                ; page size in bytes
  36                              <1> PAGE_SHIFT  equ 12                  ; page table shift count
  37                              <1> PAGE_D_SHIFT       equ 22  ; 12 + 10       ; page directory shift count
  38                              <1> PAGE_OFF    equ 0FFFh               ; 12 bit byte offset in page frame
  39                              <1> PTE_MASK    equ 03FFh               ; page table entry mask
  40                              <1> PTE_DUPLICATED  equ 200h            ; duplicated page sign (AVL bit 0)
  41                              <1> PDE_A_CLEAR equ 0F000h              ; to clear PDE attribute bits
  42                              <1> PTE_A_CLEAR equ 0F000h              ; to clear PTE attribute bits
  43                              <1> LOGIC_SECT_SIZE equ 512                     ; logical sector size
  44                              <1> ERR_MAJOR_PF       equ 0E0h                 ; major error: page fault
  45                              <1> ERR_MINOR_IM       equ 4 ;15/10/2016 (1->4); insufficient (out of) memory
  46                              <1> ERR_MINOR_PV       equ 6 ;15/10/2016 (1->4); protection violation
  47                              <1> SWP_DISK_READ_ERR     equ 40
  48                              <1> SWP_DISK_NOT_PRESENT_ERR   equ 41
  49                              <1> SWP_SECTOR_NOT_PRESENT_ERR equ 42
  50                              <1> SWP_NO_FREE_SPACE_ERR      equ 43
  51                              <1> SWP_DISK_WRITE_ERR         equ 44
  52                              <1> SWP_NO_PAGE_TO_SWAP_ERR    equ 45
  53                              <1> PTE_A_ACCESS_BIT equ 5  ; Bit 5 (accessed flag)
  54                              <1> SECTOR_SHIFT     equ 3  ; sector shift (to convert page block number)
  55                              <1> ; 12/07/2016
  56                              <1> PTE_SHARED   equ 400h               ; AVL bit 1, direct memory access bit
  57                              <1>                                     ; (Indicates that the page is not allocated
  58                              <1>                                     ; for the process, it is a shared or system
  59                              <1>                                           ; page, it must not be deallocated!)
  60                              <1> ; 14/12/2020
  61                              <1> ; (Linear Frame Buffer - video memory mark : AVL bit 1, outside M.A.T.)
  62                              <1> PDE_EXTERNAL       equ 400h         ; Page directory entry for external memory blocks
  63                              <1> PTE_EXTERNAL       equ 400h         ; Allocated kernel pages for Linear Frame Buffer
  64                              <1>                             ; (Out of memory allocation table)
  65                              <1>
  66                              <1> ;
  67                              <1> ;; Retro Unix 386 v1 - paging method/principles
  68                              <1> ;;
  69                              <1> ;; 10/10/2014
  70                              <1> ;; RETRO UNIX 386 v1 - PAGING METHOD/PRINCIPLES
  71                              <1> ;;
  72                              <1> ;; KERNEL PAGE MAP: 1 to 1 physical memory page map
  73                              <1> ;;  (virtual address = physical address)
  74                              <1> ;; KERNEL PAGE TABLES:
  75                              <1> ;;  Kernel page directory and all page tables are
  76                              <1> ;;  on memory as initialized, as equal to physical memory
  77                              <1> ;;  layout. Kernel pages can/must not be swapped out/in.
  78                              <1> ;;
  79                              <1> ;;  what for: User pages may be swapped out, when accessing
  80                              <1> ;;  a page in kernel/system mode, if it would be swapped out,
  81                              <1> ;;  kernel would have to swap it in! But it is also may be
  82                              <1> ;;  in use by a user process. (In system/kernel mode
  83                              <1> ;;  kernel can access all memory pages even if they are
  84                              <1> ;;  reserved/allocated for user processes. Swap out/in would
  85                              <1> ;;  cause conflicts.)
  86                              <1> ;;
  87                              <1> ;;  As result of these conditions,
  88                              <1> ;;  all kernel pages must be initialized as equal to
  89                              <1> ;;  physical layout for preventing page faults.
  90                              <1> ;;  Also, calling "allocate page" procedure after
  91                              <1> ;;  a page fault can cause another page fault (double fault)
  92                              <1> ;;  if all kernel page tables would not be initialized.
  93                              <1> ;;
  94                              <1> ;;  [first_page] = Beginning of users space, as offset to
  95                              <1> ;;  memory allocation table. (double word aligned)
  96                              <1> ;;
  97                              <1> ;;  [next_page] = first/next free space to be searched
  98                              <1> ;;  as offset to memory allocation table. (dw aligned)
  99                              <1> ;;
 100                              <1> ;;  [last_page] = End of memory (users space), as offset
 101                              <1> ;;  to memory allocation table. (double word aligned)
 102                              <1> ;;
 103                              <1> ;; USER PAGE TABLES:
 104                              <1> ;;  Demand paging (& 'copy on write' allocation method) ...
 105                              <1> ;;          'ready only' marked copies of the
 106                              <1> ;;          parent process's page table entries (for
 107                              <1> ;;          same physical memory).
 108                              <1> ;;          (A page will be copied to a new page after
 109                              <1> ;;           if it causes R/W page fault.)
 110                              <1> ;;
 111                              <1> ;;  Every user process has own (different)
```

```
<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>     ; 17/04/2021 - TRDOS 386 v2.0.4
                        <1>     ;       (temporary modifications)
                        <1>     ; 01/07/2015
                        <1>     ; 05/05/2015
                        <1>     ; 30/04/2015
                        <1>     ; 16/10/2014
                        <1>     ; 08/10/2014
                        <1>     ; 09/09/2014 (Retro UNIX 386 v1 - beginning)
                        <1>     ;
                        <1>     ; INPUT -> none
                        <1>     ;
                        <1>     ; OUTPUT ->
                        <1>     ;       EAX = PHYSICAL (real/flat) ADDRESS OF THE ALLOCATED PAGE
                        <1>     ;       (corresponding MEMORY ALLOCATION TABLE bit is RESET)
                        <1>     ;
                        <1>     ;       CF = 1 and EAX = 0
                        <1>     ;                  if there is not a free page to be allocated
                        <1>     ;
                        <1>     ; Modified Registers -> none (except EAX)
                        <1>     ;
 00005826 A1[88760100]  <1>     mov     eax, [free_pages]
 0000582B 21C0          <1>     and     eax, eax
 0000582D 7438          <1>     jz      short out_of_memory
                        <1>     ;
 0000582F 53            <1>     push    ebx
 00005830 51            <1>     push    ecx
                        <1>     ;
 00005831 BB00001000    <1>     mov     ebx, MEM_ALLOC_TBL   ; Memory Allocation Table offset
 00005836 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)
 00005838 031D[8C760100]<1>     add     ebx, [next_page] ; Free page searching starts from here
                        <1>                             ; next_free_page >> 5
 0000583E 030D[90760100]<1>     add     ecx, [last__page] ; Free page searching ends here
                        <1>                              ; (total_pages - 1) >> 5
                        <1> al_p_scan:
 00005844 39CB          <1>     cmp     ebx, ecx
 00005846 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>     ;
00005848 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.
0000584B 751C           <1>     jnz     short al_p_found ; ZF = 0 -> a free page has been found
                        <1>                     ;
                        <1>                     ; NOTE: a Memory Allocation Table bit
                        <1>                     ;       with value of 1 means
                        <1>                     ;       the corresponding page is free
                        <1>                     ;       (Retro UNIX 386 v1 feature only!)
0000584D 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.
00005850 EBF2           <1>         jmp     short al_p_scan
                        <1>     ;
                        <1> al_p_notfound:
00005852 81E900001000   <1>     sub     ecx, MEM_ALLOC_TBL
00005858 890D[8C760100] <1>     mov     [next_page], ecx ; next/first free page = last page
                        <1>                              ; (deallocate_page procedure will change it)
0000585E 31C0           <1>     xor     eax, eax
00005860 A3[88760100]   <1>     mov     [free_pages], eax ; 0
00005865 59             <1>     pop     ecx
00005866 5B             <1>     pop     ebx
                        <1>     ;
                        <1> ; 17/04/2021
                        <1> ; ('swap_out' procedure call is disabled as temporary)
                        <1>
                        <1> out_of_memory:
                        <1> ;   call    swap_out
                        <1> ;   jnc     short al_p_ok  ; [free_pages] = 0, re-allocation by swap_out
                        <1> ;   ;
                        <1> ;   sub     eax, eax ; 0
00005867 F9             <1>     stc
00005868 C3             <1>     retn
                        <1>
                        <1> al_p_found:
00005869 89D9           <1>     mov     ecx, ebx
0000586B 81E900001000   <1>     sub     ecx, MEM_ALLOC_TBL
00005871 890D[8C760100] <1>     mov     [next_page], ecx ; Set first free page searching start
                        <1>                              ; address/offset (to the next)
00005877 FF0D[88760100] <1>         dec     dword [free_pages] ; 1 page has been allocated (X = X-1)
                        <1>     ;
0000587D 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)
00005880 C1E103         <1>     shl     ecx, 3          ; (page block offset * 32) + page index
00005883 01C8           <1>     add     eax, ecx        ; = page number
00005885 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...
00005888 59             <1>     pop     ecx
00005889 5B             <1>     pop     ebx
                        <1> al_p_ok:
0000588A C3             <1>     retn
                        <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>     ;
0000588B E896FFFFFF     <1>     call    allocate_page
00005890 7216           <1>     jc      short mkpd_error
                        <1>     ;
00005892 A3[8C8E0100]   <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>     ;
00005897 57             <1>     push    edi
00005898 51             <1>     push    ecx
00005899 50             <1>     push    eax
0000589A B900040000     <1>     mov     ecx, PAGE_SIZE / 4
0000589F 89C7           <1>     mov     edi, eax
000058A1 31C0           <1>     xor     eax, eax
000058A3 F3AB           <1>     rep     stosd
000058A5 58             <1>     pop     eax
000058A6 59             <1>     pop     ecx
000058A7 5F             <1>     pop     edi
```

```
                                <1> mkpd_error:
                                <1> mkpt_error:
000058A8 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>     ;
000058A9 E878FFFFFF             <1>     call    allocate_page
000058AE 72F8                   <1>     jc      short mkpt_error
000058B0 E811000000             <1>     call    set_pde
000058B5 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>     ;
000058B7 E86AFFFFFF             <1>     call    allocate_page
000058BC 7207                   <1>     jc      short mkp_err
000058BE E821000000             <1>     call    set_pte
000058C3 73D2                   <1>     jnc     short clear_page ; 18/04/2015
                                <1> mkp_err:
000058C5 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>     ;
000058C6 89DA                   <1>     mov     edx, ebx
000058C8 C1EA16                 <1>     shr     edx, PAGE_D_SHIFT ; 22
000058CB C1E202                 <1>     shl     edx, 2 ; offset to page directory (1024*4)
000058CE 0315[8C8E0100]         <1>     add     edx, [u.pgdir]
                                <1>     ;
000058D4 21C0                   <1>     and     eax, eax
000058D6 7506                   <1>     jnz     short spde_1
                                <1>     ;
000058D8 8B02                   <1>     mov     eax, [edx]  ; old PDE value
                                <1>     ;test   al, 1
                                <1>     ;jz     short spde_2
000058DA 662500F0               <1>     and     ax, PDE_A_CLEAR ; 0F000h  ; clear lower 12 bits
                                <1> spde_1:
                                <1>     ;and    cx, 0FFFh
000058DE 8902                   <1>     mov     [edx], eax
000058E0 66090A                 <1>     or      [edx], cx
000058E3 C3                     <1>     retn
                                <1> ;spde_2: ; error
                                <1> ;   stc
                                <1> ;   retn
                                <1>
                                <1> set_pte:    ; Set page table entry (PTE)
                                <1>     ; 24/07/2015
                                <1>     ; 20/07/2015
                                <1>     ; 23/06/2015
                                <1>     ; 18/04/2015
                                <1>     ; 12/04/2015
                                <1>     ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
                                <1>     ;
                                <1>     ; INPUT ->
                                <1>     ;       EAX = physical page address
                                <1>     ;             (use present value if EAX = 0)
                                <1>     ;       EBX = virtual (linear) address
                                <1>     ;       ECX = page attributes (lower 12 bits)
                                <1>     ;             (higher 20 bits must be ZERO)
                                <1>     ;             (bit 0 must be 1)
                                <1>     ;       u.pgdir = page directory (physical) address
                                <1>     ; OUTPUT ->
                                <1>     ;       EAX = physical page address
                                <1>     ;       (EDX = PTE value)
                                <1>     ;       EBX = virtual address
                                <1>     ;
                                <1>     ;       CF = 1 -> error
                                <1>     ;
                                <1>     ; Modified Registers -> EAX, EDX
                                <1>     ;
```

```
000058E4 50                    <1>     push    eax
000058E5 A1[8C8E0100]          <1>     mov     eax, [u.pgdir] ; 20/07/2015
000058EA E837000000            <1>     call    get_pde
                               <1>             ; EDX = PDE address
                               <1>             ; EAX = PDE value
000058EF 5A                    <1>     pop     edx ; physical page address
000058F0 722A                  <1>     jc      short spte_err ; PDE not present
                               <1>     ;
000058F2 53                    <1>     push    ebx ; 24/07/2015
000058F3 662500F0              <1>     and     ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 bits
                               <1>                     ; EDX = PT address (physical)
000058F7 C1EB0C                <1>     shr     ebx, PAGE_SHIFT ; 12
000058FA 81E3FF030000          <1>     and     ebx, PTE_MASK  ; 03FFh
                               <1>                 ; clear higher 10 bits (PD bits)
00005900 C1E302                <1>     shl     ebx, 2   ; offset to page table (1024*4)
00005903 01C3                  <1>     add     ebx, eax
                               <1>     ;
00005905 8B03                  <1>     mov     eax, [ebx] ; Old PTE value
00005907 A801                  <1>     test    al, 1
00005909 740C                  <1>     jz      short spte_0
0000590B 09D2                  <1>     or      edx, edx
0000590D 750F                  <1>     jnz     short spte_1
0000590F 662500F0              <1>     and     ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 bits
00005913 89C2                  <1>     mov     edx, eax
00005915 EB09                  <1>     jmp     short spte_2
                               <1> spte_0:
                               <1>     ; If this PTE contains a swap (disk) address,
                               <1>     ; it can be updated by using 'swap_in' procedure
                               <1>     ; only!
00005917 21C0                  <1>     and     eax, eax
00005919 7403                  <1>     jz      short spte_1
                               <1>     ; 24/07/2015
                               <1>     ; swapped page ! (on disk)
0000591B 5B                    <1>     pop     ebx
                               <1> spte_err:
0000591C F9                    <1>     stc
0000591D C3                    <1>     retn
                               <1> spte_1:
0000591E 89D0                  <1>     mov     eax, edx
                               <1> spte_2:
00005920 09CA                  <1>     or      edx, ecx
                               <1>     ; 23/06/2015
00005922 8913                  <1>     mov     [ebx], edx ; PTE value in EDX
                               <1>     ; 24/07/2015
00005924 5B                    <1>     pop     ebx
00005925 C3                    <1>     retn
                               <1>
                               <1> get_pde:    ; Get present value of the relevant PDE
                               <1>     ; 20/07/2015
                               <1>     ; 18/04/2015
                               <1>     ; 12/04/2015
                               <1>     ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
                               <1>     ;
                               <1>     ; INPUT ->
                               <1>     ;       EBX = virtual (linear) address
                               <1>     ;       EAX = page directory (physical) address
                               <1>     ; OUTPUT ->
                               <1>     ;       EDX = Page directory entry address
                               <1>     ;       EAX = Page directory entry value
                               <1>     ;       CF = 1 -> PDE not present or invalid ?
                               <1>     ; Modified Registers -> EDX, EAX
                               <1>     ;
00005926 89DA                  <1>     mov     edx, ebx
00005928 C1EA16                <1>     shr     edx, PAGE_D_SHIFT ; 22  (12+10)
0000592B C1E202                <1>     shl     edx, 2 ; offset to page directory (1024*4)
0000592E 01C2                  <1>     add     edx, eax ; page directory address (physical)
00005930 8B02                  <1>     mov     eax, [edx]
00005932 A801                  <1>     test    al, PDE_A_PRESENT ; page table is present or not !
00005934 751F                  <1>     jnz     short gpte_retn
00005936 F9                    <1>     stc
                               <1> gpde_retn:
00005937 C3                    <1>     retn
                               <1>
                               <1> get_pte:
                               <1>             ; Get present value of the relevant PTE
                               <1>     ; 29/07/2015
                               <1>     ; 20/07/2015
                               <1>     ; 18/04/2015
                               <1>     ; 12/04/2015
                               <1>     ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
                               <1>     ;
                               <1>     ; INPUT ->
                               <1>     ;       EBX = virtual (linear) address
                               <1>     ;       EAX = page directory (physical) address
                               <1>     ; OUTPUT ->
                               <1>     ;       EDX = Page table entry address (if CF=0)
                               <1>     ;             Page directory entry address (if CF=1)
                               <1>     ;            (Bit 0 value is 0 if PT is not present)
                               <1>     ;       EAX = Page table entry value (page address)
                               <1>     ;       CF = 1 -> PDE not present or invalid ?
                               <1>     ; Modified Registers -> EAX, EDX
                               <1>     ;
00005938 E8E9FFFFFF            <1>     call    get_pde
0000593D 72F8                  <1>     jc      short gpde_retn       ; page table is not present
                               <1>     ;jnc    short gpte_1
                               <1>     ;retn
                               <1> ;gpte_1:
0000593F 662500F0              <1>     and     ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 bits
00005943 89DA                  <1>     mov     edx, ebx
00005945 C1EA0C                <1>     shr     edx, PAGE_SHIFT ; 12
00005948 81E2FF030000          <1>     and     edx, PTE_MASK  ; 03FFh
                               <1>                     ; clear higher 10 bits (PD bits)
0000594E C1E202                <1>     shl     edx, 2 ; offset from start of page table (1024*4)
00005951 01C2                  <1>     add     edx, eax
00005953 8B02                  <1>     mov     eax, [edx]
                               <1> gpte_retn:
00005955 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>     ;
00005956 56                     <1>     push    esi
00005957 51                     <1>     push    ecx
00005958 50                     <1>     push    eax
00005959 89C6                   <1>     mov     esi, eax
0000595B 31C9                   <1>     xor     ecx, ecx
                                <1>     ; The 1st PDE points to Kernel Page Table 0 (the 1st 4MB),
                                <1>     ; it must not be deallocated
0000595D 890E                   <1>     mov     [esi], ecx ; 0 ; clear PDE 0
                                <1> dapd_0:
0000595F AD                     <1>     lodsd
00005960 A801                   <1>     test    al, PDE_A_PRESENT ; bit 0, present flag (must be 1)
00005962 7409                   <1>     jz      short dapd_1
00005964 662500F0               <1>     and     ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
00005968 E812000000             <1>     call    deallocate_page_table
                                <1> dapd_1:
0000596D 41                     <1>     inc     ecx ; page directory entry index
0000596E 81F900040000           <1>     cmp     ecx, PAGE_SIZE / 4 ; 1024
00005974 72E9                   <1>     jb      short dapd_0
                                <1> dapd_2:
00005976 58                     <1>     pop     eax
00005977 E869000000             <1>     call    deallocate_page       ; deallocate the page dir's itself
0000597C 59                     <1>     pop     ecx
0000597D 5E                     <1>     pop     esi
0000597E C3                     <1>     retn
                                <1>
                                <1> deallocate_page_table:
                                <1>     ; 29/08/2023 - TRDOS 386 v2.0.6
                                <1>     ; 17/04/2021 - TRDOS 386 v2.0.4
                                <1>     ;       (temporary modifications)
                                <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>     ;
0000597F 56                     <1>     push    esi
00005980 57                     <1>     push    edi
00005981 52                     <1>     push    edx
00005982 50                     <1>     push    eax ; *
00005983 89C6                   <1>     mov     esi, eax
00005985 31FF                   <1>     xor     edi, edi ; 0
                                <1> dapt_0:
00005987 AD                     <1>     lodsd
00005988 A801                   <1>     test    al, PTE_A_PRESENT ; bit 0, present flag (must be 1)
0000598A 744C                   <1>     jz      short dapt_1
                                <1>     ;
0000598C A802                   <1>     test    al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
                                <1>                               ; (must be 1)
0000598E 753D                   <1>     jnz     short dapt_3
                                <1>     ; Read only -duplicated- page (belongs to a parent or a child)
                                <1>         test    ax, PTE_DUPLICATED ; Was this page duplicated
                                <1>                                    ; as child's page ?
00005994 7442                   <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)
00005996 53                     <1>     push    ebx
00005997 51                     <1>     push    ecx
                                <1>     ;shl    cx, 2 ; *4
                                <1>     ; 29/08/2023
00005998 C1E102                 <1>     shl     ecx, 2
0000599B 01CB                   <1>     add     ebx, ecx ; PDE offset (for the parent)
0000599D 8B0B                   <1>     mov     ecx, [ebx]
0000599F F6C101                 <1>     test    cl, PDE_A_PRESENT ; present (valid) or not ?
000059A2 7427                   <1>     jz      short dapt_2   ; parent process does not use this page
000059A4 6681E100F0             <1>     and     cx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
                                <1>     ; EDI = page table entry index (0-1023)
000059A9 89FA                   <1>     mov     edx, edi
                                <1>     ;shl    dx, 2 ; *4
                                <1>     ; 29/08/2023
000059AB C1E202                 <1>     shl     edx, 2
000059AE 01CA                   <1>     add     edx, ecx ; PTE offset (for the parent)
000059B0 8B1A                   <1>     mov     ebx, [edx]
000059B2 F6C301                 <1>     test    bl, PTE_A_PRESENT ; present or not ?
000059B5 7414                   <1>     jz      short dapt_2   ; parent process does not use this page
000059B7 662500F0               <1>     and     ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
000059BB 6681E300F0             <1>     and     bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
000059C0 39D8                   <1>     cmp     eax, ebx       ; parent's and child's pages are same ?
000059C2 7507                   <1>     jne     short dapt_2   ; not same page
                                <1>                            ; deallocate the child's page
000059C4 800A02                 <1>         or      byte [edx], PTE_A_WRITE ; convert to writable page (parent)
000059C7 59                     <1>     pop     ecx
000059C8 5B                     <1>     pop     ebx
000059C9 EB0D                   <1>     jmp     short dapt_4
                                <1>
                                <1> ; 17/04/2021
                                <1> ; ('dapt_1' is disabled as temporary)
                                <1> ;
                                <1> ;dapt_1:
                                <1> ;   or      eax, eax       ; swapped page ?
                                <1> ;   jz      short dapt_5   ; no
                                <1> ;                          ; yes
                                <1> ;   shr     eax, 1
                                <1> ;   call    unlink_swap_block ; Deallocate swapped page block
                                <1> ;                             ; on the swap disk (or in file)
                                <1> ;   jmp     short dapt_5
                                <1> dapt_2:
000059CB 59                     <1>     pop     ecx
000059CC 5B                     <1>     pop     ebx
                                <1> dapt_3:
                                <1>     ; 12/07/2016
000059CD 66A90004               <1>     test    ax, PTE_SHARED ; shared or direct memory access indicator
000059D1 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
000059D3 E80D000000             <1>     call    deallocate_page ; set the mem allocation bit of this page
```

```
                                <1> dapt_4:
                                <1>     ; 20/10/2023
                                <1>     ; PTE clearing here was/is not necessary
                                <1>     ; because this page table is being deallocated
                                <1>     ; and it's PTEs will be ineffective.
                                <1> 
                                <1>     ;mov    dword [esi-4], 0 ; clear/reset PTE (child, dupl. as parent)
                                <1> 
                                <1> dapt_1:     ; 17/04/2021 (temporary)
                                <1> dapt_5:
000059D8 47                     <1>     inc     edi ; page table entry index
000059D9 81FF00040000           <1>     cmp     edi, PAGE_SIZE / 4 ; 1024
000059DF 72A6                   <1>     jb      short dapt_0
                                <1>     ;
000059E1 58                     <1>     pop     eax ; *
000059E2 5A                     <1>     pop     edx
000059E3 5F                     <1>     pop     edi
000059E4 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>     ;
000059E5 53                     <1>     push    ebx
000059E6 52                     <1>     push    edx
                                <1>     ;
000059E7 C1E80C                 <1>     shr     eax, PAGE_SHIFT       ; shift physical address to
                                <1>                                  ; 12 bits right
                                <1>                                  ; to get page number
000059EA 89C2                   <1>     mov     edx, eax
                                <1>     ; 15/09/2015
000059EC C1EA03                 <1>     shr     edx, 3               ; to get offset to M.A.T.
                                <1>                                  ; (1 allocation bit = 1 page)
                                <1>                                  ; (1 allocation bytes = 8 pages)
000059EF 80E2FC                 <1>     and     dl, 0FCh             ; clear lower 2 bits
                                <1>                                  ; (to get 32 bit position)
                                <1>     ;
000059F2 BB00001000             <1>     mov     ebx, MEM_ALLOC_TBL    ; Memory Allocation Table address
000059F7 01D3                   <1>     add     ebx, edx
000059F9 83E01F                 <1>     and     eax, 1Fh             ; lower 5 bits only
                                <1>                                  ; (allocation bit position)
000059FC 3B15[8C760100]         <1>     cmp     edx, [next_page]      ; is the new free page address lower
                                <1>                                  ; than the address in 'next_page' ?
                                <1>                                  ; (next/first free page value)
00005A02 7306                   <1>     jnb     short dap_1          ; no
00005A04 8915[8C760100]         <1>     mov     [next_page], edx      ; yes
                                <1> dap_1:
00005A0A 0FAB03                 <1>     bts     [ebx], eax           ; unlink/release/deallocate page
                                <1>                                  ; set relevant bit to 1.
                                <1>                                  ; set CF to the previous bit value
                                <1>     ;cmc                         ; complement carry flag
                                <1>     ;jc     short dap_2          ; do not increase free_pages count
                                <1>                                  ; if the page is already deallocated
                                <1>                                  ; before.
00005A0D FF05[88760100]         <1>         inc     dword [free_pages]
                                <1> dap_2:
00005A13 5A                     <1>     pop     edx
00005A14 5B                     <1>     pop     ebx
00005A15 C3                     <1>     retn
                                <1> 
                                <1> ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                                <1> ;;                                                                  ;;
                                <1> ;; Copyright (C) KolibriOS team 2004-2012. All rights reserved.    ;;
                                <1> ;; Distributed under terms of the GNU General Public License       ;;
                                <1> ;;                                                                  ;;
                                <1> ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                                <1> 
                                <1> ;;$Revision: 5057 $
                                <1> 
                                <1> 
                                <1> ;;align 4
                                <1> ;;proc alloc_page
                                <1> 
                                <1> ;;        pushfd
                                <1> ;;        cli
                                <1> ;;        push    ebx
                                <1> ;;;//-
                                <1> ;;        cmp     [pg_data.pages_free], 1
                                <1> ;;        jle     .out_of_memory
                                <1> ;;;//-
                                <1> ;;
                                <1> ;;        mov     ebx, [page_start]
                                <1> ;;        mov     ecx, [page_end]
                                <1> ;;.l1:
                                <1> ;;        bsf     eax, [ebx];
                                <1> ;;        jnz     .found
                                <1> ;;        add     ebx, 4
                                <1> ;;        cmp     ebx, ecx
                                <1> ;;        jb      .l1
                                <1> ;;        pop     ebx
                                <1> ;;        popfd
                                <1> ;;        xor     eax, eax
                                <1> ;;        ret
                                <1> ;;.found:
                                <1> ;;;//-
                                <1> ;;        dec     [pg_data.pages_free]
                                <1> ;;        jz      .out_of_memory
                                <1> ;;;//-
                                <1> ;;        btr     [ebx], eax
                                <1> ;;        mov     [page_start], ebx
                                <1> ;;        sub     ebx, sys_pgmap
                                <1> ;;        lea     eax, [eax+ebx*8]
                                <1> ;;        shl     eax, 12
                                <1> ;;;//-       dec [pg_data.pages_free]
                                <1> ;;        pop     ebx
                                <1> ;;        popfd
                                <1> ;;        ret
                                <1> ;;;//-
```

```
                            <1> ;;.out_of_memory:
                            <1> ;;      mov     [pg_data.pages_free], 1
                            <1> ;;      xor     eax, eax
                            <1> ;;      pop     ebx
                            <1> ;;      popfd
                            <1> ;;      ret
                            <1> ;;;//-
                            <1> ;;endp
                            <1>
                            <1> duplicate_page_dir:
                            <1>     ; 21/09/2015
                            <1>     ; 31/08/2015
                            <1>     ; 20/07/2015
                            <1>     ; 28/04/2015
                            <1>     ; 27/04/2015
                            <1>     ; 18/04/2015
                            <1>     ; 12/04/2015
                            <1>     ; 18/10/2014
                            <1>     ; 16/10/2014 (Retro UNIX 386 v1 - beginning)
                            <1>     ;
                            <1>     ; INPUT ->
                            <1>     ;       [u.pgdir] = PHYSICAL (real/flat) ADDRESS of the parent's
                            <1>     ;                   page directory.
                            <1>     ; OUTPUT ->
                            <1>     ;       EAX =  PHYSICAL (real/flat) ADDRESS of the child's
                            <1>     ;              page directory.
                            <1>     ;       (New page directory with new page table entries.)
                            <1>     ;       (New page tables with read only copies of the parent's
                            <1>     ;       pages.)
                            <1>     ;       EAX = 0 -> Error (CF = 1)
                            <1>     ;
                            <1>     ; Modified Registers -> none (except EAX)
                            <1>     ;
00005A16 E80BFEFFFF         <1>     call    allocate_page
00005A1B 723E               <1>     jc      short dpd_err
                            <1>     ;
00005A1D 55                 <1>     push    ebp ; 20/07/2015
00005A1E 56                 <1>     push    esi
00005A1F 57                 <1>     push    edi
00005A20 53                 <1>     push    ebx
00005A21 51                 <1>     push    ecx
00005A22 8B35[8C8E0100]     <1>     mov     esi, [u.pgdir]
00005A28 89C7               <1>     mov     edi, eax
00005A2A 50                 <1>     push    eax ; save child's page directory address
                            <1>     ; 31/08/2015
                            <1>     ; copy PDE 0 from the parent's page dir to the child's page dir
                            <1>     ; (use same system space for all user page tables)
00005A2B A5                 <1>     movsd
00005A2C BD00004000         <1>     mov     ebp, 1024*4096 ; pass the 1st 4MB (system space)
00005A31 B9FF030000         <1>     mov     ecx, (PAGE_SIZE / 4) - 1 ; 1023
                            <1> dpd_0:
00005A36 AD                 <1>     lodsd
                            <1>     ;or     eax, eax
                            <1>         ;jnz     short dpd_1
00005A37 A801               <1>     test    al, PDE_A_PRESENT ;  bit 0 =  1
00005A39 7508               <1>     jnz     short dpd_1
                            <1>     ; 20/07/2015 (virtual address at the end of the page table)
00005A3B 81C500004000       <1>     add     ebp, 1024*4096 ; page size * PTE count
00005A41 EB0F               <1>     jmp     short dpd_2
                            <1> dpd_1:
00005A43 662500F0           <1>     and     ax, PDE_A_CLEAR ; 0F000h ; clear attribute bits
00005A47 89C3               <1>     mov     ebx, eax
                            <1>     ; EBX = Parent's page table address
00005A49 E81F000000         <1>     call    duplicate_page_table
00005A4E 720C               <1>     jc      short dpd_p_err
                            <1>     ; EAX = Child's page table address
00005A50 0C07               <1>     or      al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER
                            <1>                     ; set bit 0, bit 1 and bit 2 to 1
                            <1>                     ; (present, writable, user)
                            <1> dpd_2:
00005A52 AB                 <1>     stosd
00005A53 E2E1               <1>     loop    dpd_0
                            <1>     ;
00005A55 58                 <1>     pop     eax  ; restore child's page directory address
                            <1> dpd_3:
00005A56 59                 <1>     pop     ecx
00005A57 5B                 <1>     pop     ebx
00005A58 5F                 <1>     pop     edi
00005A59 5E                 <1>     pop     esi
00005A5A 5D                 <1>     pop     ebp ; 20/07/2015
                            <1> dpd_err:
00005A5B C3                 <1>     retn
                            <1> dpd_p_err:
                            <1>     ; release the allocated pages missing (recover free space)
00005A5C 58                 <1>     pop     eax  ; the new page directory address (physical)
00005A5D 8B1D[8C8E0100]     <1>     mov     ebx, [u.pgdir] ; parent's page directory address
00005A63 E8EEFEFFFF         <1>     call    deallocate_page_dir
00005A68 29C0               <1>     sub     eax, eax ; 0
00005A6A F9                 <1>     stc
00005A6B EBE9               <1>     jmp     short dpd_3
                            <1>
                            <1> duplicate_page_table:
                            <1>     ; 17/04/2021 - TRDOS 386 v2.0.4
                            <1>     ;       (temporary modifications)
                            <1>     ; 20/02/2017
                            <1>     ; 21/09/2015
                            <1>     ; 20/07/2015
                            <1>     ; 05/05/2015
                            <1>     ; 28/04/2015
                            <1>     ; 27/04/2015
                            <1>     ; 18/04/2015
                            <1>     ; 18/10/2014
                            <1>     ; 16/10/2014 (Retro UNIX 386 v1 - beginning)
                            <1>     ;
                            <1>     ; INPUT ->
                            <1>     ;       EBX = PHYSICAL (real/flat) ADDRESS of the parent's page table.
                            <1>     ;        20/02/2017
                            <1>     ;       EBP = Linear address of the page (from 'duplicate_page_dir')
                            <1>     ;             (Linear address = CORE + user's virtual address)
                            <1>     ; OUTPUT ->
                            <1>     ;       EAX = PHYSICAL (real/flat) ADDRESS of the child's page table.
                            <1>     ;             (with 'read only' attribute of page table entries)
                            <1>     ;       20/02/2017
                            <1>     ;       EBP = Next linear page address (for 'duplicate_page_dir')
                            <1>     ;
                            <1>     ;       CF = 1 -> error
                            <1>     ;
                            <1>     ; Modified Registers -> EBP (except EAX)
                            <1>     ;
00005A6D E8B4FDFFFF         <1>     call    allocate_page
00005A72 7258               <1>     jc      short dpt_err
                            <1>     ;
00005A74 50                 <1>     push    eax ; *
00005A75 56                 <1>     push    esi
```

```
1104 00005A76 57                <1>     push    edi
1105 00005A77 52                <1>     push    edx
1106 00005A78 51                <1>     push    ecx
1107                            <1>     ;
1108 00005A79 89DE              <1>     mov     esi, ebx
1109 00005A7B 89C7              <1>     mov     edi, eax
1110 00005A7D 89C2              <1>     mov     edx, eax
1111 00005A7F 81C200100000      <1>     add     edx, PAGE_SIZE
1112                            <1> dpt_0:
1113 00005A85 AD                <1>     lodsd
1114 00005A86 21C0              <1>     and     eax, eax
1115 00005A88 7432              <1>     jz      short dpt_3
1116 00005A8A A801              <1>     test    al, PTE_A_PRESENT ;  bit 0 =  1
1117                            <1>     ; 17/04/2021 (temporary)
1118                            <1>     ;jnz    short dpt_1
1119 00005A8C 7439              <1>     jz      short dpt_p_err
1120                            <1>
1121                            <1> ; 17/04/2021
1122                            <1> ; ('reload_page' procedure call is disabled as temporary)
1123                            <1> ;
1124                            <1> ;   ; 20/07/2015
1125                            <1> ;   ; ebp = virtual (linear) address of the memory page
1126                            <1> ;   call    reload_page ; 28/04/2015
1127                            <1> ;   jc      short dpt_p_err
1128                            <1> dpt_1:
1129                            <1>     ; 21/09/2015
1130 00005A8E 89C1              <1>     mov     ecx, eax
1131 00005A90 662500F0          <1>     and     ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
1132 00005A94 F6C102            <1>     test    cl, PTE_A_WRITE ; writable page ?
1133 00005A97 751A              <1>     jnz     short dpt_2
1134                            <1>     ; Read only (parent) page
1135                            <1>     ;       - there is a third process which uses this page -
1136                            <1>     ; Allocate a new page for the child process
1137 00005A99 E888FDFFFF        <1>     call    allocate_page
1138 00005A9E 7227              <1>     jc      short dpt_p_err
1139 00005AA0 57                <1>     push    edi
1140 00005AA1 56                <1>     push    esi
1141 00005AA2 89CE              <1>     mov     esi, ecx
1142 00005AA4 89C7              <1>     mov     edi, eax
1143 00005AA6 B900040000        <1>     mov     ecx, PAGE_SIZE/4
1144 00005AAB F3A5              <1>     rep     movsd   ; copy page (4096 bytes)
1145 00005AAD 5E                <1>     pop     esi
1146 00005AAE 5F                <1>     pop     edi
1147                            <1>     ;
1148                            <1>
1149                            <1> ; 17/04/2021
1150                            <1> ; ('add_to_swap_queue' procedure call is disabled as temporary)
1151                            <1> ;
1152                            <1> ;   push    ebx
1153                            <1> ;   push    eax
1154                            <1> ;   ; 20/07/2015
1155                            <1> ;   mov     ebx, ebp
1156                            <1> ;   ; ebx = virtual (linear) address of the memory page
1157                            <1> ;   call    add_to_swap_queue
1158                            <1> ;   pop     eax
1159                            <1> ;   pop     ebx
1160                            <1>
1161                            <1>     ; 21/09/2015
1162 00005AAF 0C07              <1>     or      al, PTE_A_USER+PTE_A_WRITE+PTE_A_PRESENT
1163                            <1>             ; user + writable + present page
1164 00005AB1 EB09              <1>     jmp     short dpt_3
1165                            <1> dpt_2:
1166                            <1>     ;or     ax, PTE_A_USER+PTE_A_PRESENT
1167 00005AB3 0C05              <1>     or      al, PTE_A_USER+PTE_A_PRESENT
1168                            <1>                 ; (read only page!)
1169 00005AB5 8946FC            <1>     mov     [esi-4], eax ; update parent's PTE
1170 00005AB8 660D0002          <1>     or      ax, PTE_DUPLICATED  ; (read only page & duplicated PTE!)
1171                            <1> dpt_3:
1172 00005ABC AB                <1>     stosd  ; EDI points to child's PTE
1173                            <1>     ;
1174 00005ABD 81C500100000      <1>     add     ebp, 4096 ; 20/07/2015 (next page)
1175                            <1>     ;
1176 00005AC3 39D7              <1>     cmp     edi, edx
1177 00005AC5 72BE              <1>     jb      short dpt_0
1178                            <1> dpt_p_err:
1179 00005AC7 59                <1>     pop     ecx
1180 00005AC8 5A                <1>     pop     edx
1181 00005AC9 5F                <1>     pop     edi
1182 00005ACA 5E                <1>     pop     esi
1183 00005ACB 58                <1>     pop     eax ; *
1184                            <1> dpt_err:
1185 00005ACC C3                <1>     retn
1186                            <1>
1187                            <1> page_fault_handler:       ; CPU EXCEPTION 0Eh (14) : Page Fault !
1188                            <1>     ; 29/11/2023 - TRDOS 386 v2.0.7
1189                            <1>     ; 17/04/2021 - TRDOS 386 v2.0.4
1190                            <1>     ;       (temporary modifications)
1191                            <1>     ; 21/09/2015
1192                            <1>     ; 19/09/2015
1193                            <1>     ; 17/09/2015
1194                            <1>     ; 28/08/2015
1195                            <1>     ; 20/07/2015
1196                            <1>     ; 28/06/2015
1197                            <1>     ; 03/05/2015
1198                            <1>     ; 30/04/2015
1199                            <1>     ; 18/04/2015
1200                            <1>     ; 12/04/2015
1201                            <1>     ; 30/10/2014
1202                            <1>     ; 11/09/2014
1203                            <1>     ; 10/09/2014 (Retro UNIX 386 v1 - beginning)
1204                            <1>     ;
1205                            <1>     ; Note: This is not an interrupt/exception handler.
1206                            <1>     ;       This is a 'page fault remedy' subroutine
1207                            <1>     ;       which will be called by standard/uniform
1208                            <1>     ;       exception handler.
1209                            <1>     ;
1210                            <1>     ; INPUT ->
1211                            <1>     ;       [error_code] = 32 bit ERROR CODE (lower 5 bits are valid)
1212                            <1>     ;
1213                            <1>     ;       cr2 = the virtual (linear) address
1214                            <1>     ;             which has caused to page fault (19/09/2015)
1215                            <1>     ;
1216                            <1>     ; OUTPUT ->
1217                            <1>     ;       (corresponding PAGE TABLE ENTRY is mapped/set)
1218                            <1>     ;       EAX = 0 -> no error
1219                            <1>     ;       EAX > 0 -> error code in EAX (also CF = 1)
1220                            <1>     ;
1221                            <1>     ; Modified Registers -> none (except EAX)
1222                            <1>     ;
1223                            <1>       ;
1224                            <1>       ; ERROR CODE:
1225                            <1>     ;       31  .....     4   3   2   1   0
1226                            <1>     ;       +---+-- --+---+---+---+---+---+
1227                            <1>     ;       |   Reserved  | I | R | U | W | P |
```

```
1228                                    <1>     ;      +---+-- --+---+---+---+---+---+---+
1229                                    <1>     ;
1230                                    <1>     ; P : PRESENT -       When set, the page fault was caused by
1231                                    <1>     ;               a page-protection violation. When not set,
1232                                    <1>     ;               it was caused by a non-present page.
1233                                    <1>     ; W : WRITE   -       When set, the page fault was caused by
1234                                    <1>     ;               a page write. When not set, it was caused
1235                                    <1>     ;               by a page read.
1236                                    <1>     ; U : USER    -       When set, the page fault was caused
1237                                    <1>     ;               while CPL = 3.
1238                                    <1>     ;               This does not necessarily mean that
1239                                    <1>     ;               the page fault was a privilege violation.
1240                                    <1>     ; R : RESERVD -       When set, the page fault was caused by
1241                                    <1>     ;     WRITE   reading a 1 in a reserved field.
1242                                    <1>     ; I : INSTRUC -       When set, the page fault was caused by
1243                                    <1>     ;     FETCH   an instruction fetch
1244                                    <1>     ;
1245                                    <1>     ;; x86 (32 bit) VIRTUAL ADDRESS TRANSLATION
1246                                    <1>     ;  31               22                  12 11                    0
1247                                    <1>     ; +------------------+------------------+-----------------------+
1248                                    <1>           ; | PAGE DIR. ENTRY # | PAGE TAB. ENTRY # |        OFFSET         |
1249                                    <1>           ; +-------------------+-------------------+-----------------------+
1250                                    <1>     ;
1251                                    <1>
1252                                    <1>     ;; CR3 REGISTER (Control Register 3)
1253                                    <1>     ;  31                                    12             5 4 3 2   0
1254                                    <1>     ; +----------------------------------------+-------------+---+-----+
1255                                    <1>           ; |                                        |             |P|P|     |
1256                                    <1>           ; |   PAGE DIRECTORY TABLE BASE ADDRESS    |  reserved   |C|W|rsvrd|
1257                                    <1>           ; |                                        |         |D|T|     |
1258                                    <1>     ; +----------------------------------------+-------------+---+-----+
1259                                    <1>     ;
1260                                    <1>     ;      PWT    - WRITE THROUGH
1261                                    <1>     ;      PCD    - CACHE DISABLE
1262                                    <1>     ;
1263                                    <1>     ;
1264                                    <1>     ;; x86 PAGE DIRECTORY ENTRY (4 KByte Page)
1265                                    <1>     ;  31                                    12 11  9 8 7 6 5 4 3 2 1 0
1266                                    <1>     ; +----------------------------------------+-----+---+-+-+-+---+-+-+-+
1267                                    <1>           ; |                                        |     | | | | |P|P|U|R| |
1268                                    <1>           ; |     PAGE TABLE BASE ADDRESS 31..12     | AVL |G|0|D|A|C|W|/|/|P|
1269                                    <1>           ; |                                        |     | | | | |D|T|S|W| |
1270                                    <1>     ; +----------------------------------------+-----+---+-+-+-+---+-+-+-+
1271                                    <1>     ;
1272                                    <1>        ;       P      - PRESENT
1273                                    <1>        ;       R/W    - READ/WRITE
1274                                    <1>        ;       U/S    - USER/SUPERVISOR
1275                                    <1>     ;       PWT    - WRITE THROUGH
1276                                    <1>     ;       PCD    - CACHE DISABLE
1277                                    <1>     ;       A      - ACCESSED
1278                                    <1>        ;       D      - DIRTY (IGNORED)
1279                                    <1>     ;       PAT    - PAGE ATTRIBUTE TABLE INDEX (CACHE BEHAVIOR)
1280                                    <1>     ;       G      - GLOBAL       (IGNORED)
1281                                    <1>        ;       AVL    - AVAILABLE FOR SYSTEMS PROGRAMMER USE
1282                                    <1>     ;
1283                                    <1>     ;
1284                                    <1>     ;; x86 PAGE TABLE ENTRY (4 KByte Page)
1285                                    <1>     ;  31                                    12 11  9 8 7 6 5 4 3 2 1 0
1286                                    <1>     ; +----------------------------------------+-----+---+-+-+-+---+-+-+-+
1287                                    <1>           ; |                                        |     | |P| | |P|P|U|R| |
1288                                    <1>           ; |     PAGE FRAME BASE ADDRESS 31..12     | AVL |G|A|D|A|C|W|/|/|P|
1289                                    <1>           ; |                                        |     | |T| | |D|T|S|W| |
1290                                    <1>     ; +----------------------------------------+-----+---+-+-+-+---+-+-+-+
1291                                    <1>     ;
1292                                    <1>        ;       P      - PRESENT
1293                                    <1>        ;       R/W    - READ/WRITE
1294                                    <1>        ;       U/S    - USER/SUPERVISOR
1295                                    <1>     ;       PWT    - WRITE THROUGH
1296                                    <1>     ;       PCD    - CACHE DISABLE
1297                                    <1>     ;       A      - ACCESSED
1298                                    <1>        ;       D      - DIRTY
1299                                    <1>     ;       PAT    - PAGE ATTRIBUTE TABLE INDEX (CACHE BEHAVIOR)
1300                                    <1>     ;       G      - GLOBAL
1301                                    <1>        ;       AVL    - AVAILABLE FOR SYSTEMS PROGRAMMER USE
1302                                    <1>     ;
1303                                    <1>     ;
1304                                    <1>     ;; 80386 PAGE TABLE ENTRY (4 KByte Page)
1305                                    <1>     ;  31                                    12 11  9 8 7 6 5 4 3 2 1 0
1306                                    <1>     ; +----------------------------------------+-----+-+-+-+-+---+-+-+-+
1307                                    <1>           ; |                                        |     | | | | | |U|R| |
1308                                    <1>           ; |     PAGE FRAME BASE ADDRESS 31..12     | AVL |0|0|D|A|0|0|/|/|P|
1309                                    <1>           ; |                                        |     | | | | | | |S|W| |
1310                                    <1>           ; +----------------------------------------+-----+-+-+-+-+---+-+-+-+
1311                                    <1>     ;
1312                                    <1>        ;       P      - PRESENT
1313                                    <1>        ;       R/W    - READ/WRITE
1314                                    <1>        ;       U/S    - USER/SUPERVISOR
1315                                    <1>        ;       D      - DIRTY
1316                                    <1>        ;       AVL    - AVAILABLE FOR SYSTEMS PROGRAMMER USE
1317                                    <1>     ;
1318                                    <1>        ;       NOTE: 0 INDICATES INTEL RESERVED. DO NOT DEFINE.
1319                                    <1>     ;
1320                                    <1>     ;
1321                                    <1>     ;; Invalid Page Table Entry
1322                                    <1>     ; 31                                                          1 0
1323                                    <1>           ; +-------------------------------------------------------------+-+
1324                                    <1>           ; |                                                             | |
1325                                    <1>           ; |                            AVAILABLE                        |0|
1326                                    <1>           ; |                                                             | |
1327                                    <1>           ; +-------------------------------------------------------------+-+
1328                                    <1>     ;
1329                                    <1>
1330 00005ACD 53                       <1>     push    ebx
1331 00005ACE 52                       <1>     push    edx
1332 00005ACF 51                       <1>     push    ecx
1333                                    <1>     ;
1334                                    <1>     ; 21/09/2015 (debugging)
1335 00005AD0 FF05[A48E0100]           <1>     inc     dword [u.pfcount] ; page fault count for running process
1336 00005AD6 FF05[50900100]           <1>     inc     dword [PF_Count] ; total page fault count
1337                                    <1>     ; 28/06/2015
1338                                    <1>     ;mov    edx, [error_code] ; Lower 5 bits are valid
1339 00005ADC 8A15[48900100]           <1>     mov     dl, [error_code]
1340                                    <1>     ;
1341 00005AE2 F6C201                   <1>     test    dl, 1   ; page fault was caused by a non-present page
1342                                    <1>                     ; sign
1343 00005AE5 7425                     <1>     jz      short pfh_alloc_np
1344                                    <1>     ;
1345                                    <1>     ; If it is not a 'write on read only page' type page fault
1346                                    <1>     ; major page fault error with minor reason must be returned without
1347                                    <1>     ; fixing the problem. 'sys_exit with error' will be needed
1348                                    <1>     ; after return here!
1349                                    <1>     ; Page fault will be remedied, by copying page contents
1350                                    <1>     ; to newly allocated page with write permission;
1351                                    <1>     ; sys_fork -> sys_exec -> copy on write, demand paging method is
```

```
1352                            <1>     ; used for working with minimum possible memory usage.
1353                            <1>     ; sys_fork will duplicate page directory and tables of parent
1354                            <1>     ; process with 'read only' flag. If the child process attempts to
1355                            <1>     ; write on these read only pages, page fault will be directed here
1356                            <1>     ; for allocating a new page with same data/content.
1357                            <1>     ;
1358                            <1>     ; IMPORTANT : Retro UNIX 386 v1 (and SINGLIX and TR-DOS)
1359                            <1>     ; will not force to separate CODE and DATA space
1360                            <1>     ; in a process/program...
1361                            <1>     ; CODE segment/section may contain DATA!
1362                            <1>     ; It is flat, smoth and simplest programming method already as in
1363                            <1>     ; Retro UNIX 8086 v1 and MS-DOS programs.
1364                            <1>     ;
1365 00005AE7 F6C202             <1>     test   dl, 2   ; page fault was caused by a page write
1366                            <1>                    ; sign
1367 00005AEA 7418             <1>         jz      short pfh_p_err
1368                            <1>     ; 31/08/2015
1369 00005AEC F6C204             <1>     test   dl, 4   ; page fault was caused while CPL = 3 (user mode)
1370                            <1>                    ; sign.  (U+W+P = 4+2+1 = 7)
1371 00005AEF 7413             <1>         jz  short pfh_pv_err
1372                            <1>     ;
1373                            <1>     ; make a new page and copy the parent's page content
1374                            <1>     ; as the child's new page content
1375                            <1>     ;
1376 00005AF1 0F20D3             <1>     mov    ebx, cr2 ; CR2 contains the linear address
1377                            <1>                     ; which has caused to page fault
1378 00005AF4 E87C000000         <1>     call   copy_page
1379 00005AF9 7202             <1>         jc  short pfh_im_err ; insufficient memory
1380                            <1>     ;
1381 00005AFB EB72             <1>         jmp     pfh_cpp_ok
1382                            <1>
1383                            <1>     ; 29/11/2023
1384                            <1> pfh_im_err:
1385 00005AFD B8E4000000         <1>     mov    eax, ERR_MAJOR_PF + ERR_MINOR_IM ; Error code in AX
1386                            <1>                     ; Major (Primary) Error: Page Fault
1387                            <1>                     ; Minor (Secondary) Error: Insufficient Memory !
1388 00005B02 EB6D             <1>     jmp    short pfh_err_retn
1389                            <1>
1390                            <1>     ; 29/11/2023
1391                            <1> pfh_p_err: ; 09/03/2015
1392                            <1> pfh_pv_err:
1393                            <1>     ; Page fault was caused by a protection-violation
1394 00005B04 B8E6000000         <1>     mov    eax, ERR_MAJOR_PF + ERR_MINOR_PV ; Error code in AX
1395                            <1>                     ; Major (Primary) Error: Page Fault
1396                            <1>                     ; Minor (Secondary) Error: Protection violation !
1397 00005B09 F9               <1>     stc
1398 00005B0A EB65             <1>     jmp    short pfh_err_retn
1399                            <1>
1400                            <1> pfh_alloc_np:
1401 00005B0C E815FDFFFF         <1>     call   allocate_page  ; (allocate a new page)
1402 00005B11 72EA             <1>         jc      pfh_im_err; 'insufficient memory' error
1403                            <1> pfh_chk_cpl:
1404                            <1>     ; EAX = Physical (base) address of the allocated (new) page
1405                            <1>             ; (Lower 12 bits are ZERO, because
1406                            <1>             ;       the address is on a page boundary)
1407 00005B13 80E204             <1>     and    dl, 4   ; CPL = 3 ?
1408 00005B16 7505             <1>     jnz    short pfh_um
1409                            <1>                     ; Page fault handler for kernel/system mode (CPL=0)
1410 00005B18 0F20DB             <1>     mov    ebx, cr3 ; CR3 (Control Register 3) contains physical address
1411                            <1>                      ; of the current/active page directory
1412                            <1>                      ; (Always kernel/system mode page directory, here!)
1413                            <1>                      ; Note: Lower 12 bits are 0. (page boundary)
1414 00005B1B EB06             <1>     jmp    short pfh_get_pde
1415                            <1>     ;
1416                            <1> pfh_um:                     ; Page fault handler for user/appl. mode (CPL=3)
1417 00005B1D 8B1D[8C8E0100]     <1>     mov    ebx, [u.pgdir] ; Page directory of current/active process
1418                            <1>                     ; Physical address of the USER's page directory
1419                            <1>                     ; Note: Lower 12 bits are 0. (page boundary)
1420                            <1> pfh_get_pde:
1421 00005B23 80CA03             <1>     or     dl, 3   ; USER + WRITE + PRESENT or SYSTEM + WRITE + PRESENT
1422 00005B26 0F20D1             <1>     mov    ecx, cr2 ; CR2 contains the virtual address
1423                            <1>                      ; which has been caused to page fault
1424                            <1>                      ;
1425 00005B29 C1E914             <1>     shr    ecx, 20  ; shift 20 bits right
1426 00005B2C 80E1FC             <1>     and    cl, 0FCh ; mask lower 2 bits to get PDE offset
1427                            <1>     ;
1428 00005B2F 01CB               <1>     add    ebx, ecx ; now, EBX points to the relevant page dir entry
1429 00005B31 8B0B               <1>     mov    ecx, [ebx] ; physical (base) address of the page table
1430 00005B33 F6C101             <1>     test   cl, 1    ; check bit 0 is set (1) or not (0).
1431 00005B36 740B               <1>     jz     short pfh_set_pde ; Page directory entry is not valid,
1432                            <1>                              ; set/validate page directory entry
1433 00005B38 6681E100F0         <1>     and    cx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
1434 00005B3D 89CB               <1>     mov    ebx, ecx ; Physical address of the page table
1435 00005B3F 89C1               <1>     mov    ecx, eax ; new page address (physical)
1436 00005B41 EB16               <1>     jmp    short pfh_get_pte
1437                            <1> pfh_set_pde:
1438                            <1>     ;; NOTE: Page directories and page tables never be swapped out!
1439                            <1>     ;;       (So, we know this PDE is empty or invalid)
1440                            <1>     ;
1441 00005B43 08D0               <1>     or     al, dl   ; lower 3 bits are used as U/S, R/W, P flags
1442 00005B45 8903               <1>     mov    [ebx], eax ; Let's put the new page directory entry here !
1443 00005B47 30C0               <1>     xor    al, al   ; clear lower (3..8) bits
1444 00005B49 89C3               <1>     mov    ebx, eax
1445 00005B4B E8D6FCFFFF         <1>     call   allocate_page   ; (allocate a new page)
1446 00005B50 72AB               <1>     jc     short pfh_im_err   ; 'insufficient memory' error
1447                            <1> pfh_spde_1:
1448                            <1>     ; EAX = Physical (base) address of the allocated (new) page
1449 00005B52 89C1               <1>     mov    ecx, eax
1450 00005B54 E83EFDFFFF         <1>     call   clear_page ; Clear page content
1451                            <1> pfh_get_pte:
1452 00005B59 0F20D0             <1>     mov    eax, cr2 ; virtual address
1453                            <1>                     ; which has been caused to page fault
1454 00005B5C 89C7               <1>     mov    edi, eax ; 20/07/2015
1455 00005B5E C1E80C             <1>     shr    eax, 12  ; shift 12 bit right to get
1456                            <1>                     ; higher 20 bits of the page fault address
1457 00005B61 25FF030000         <1>     and    eax, 3FFh ; mask PDE# bits, the result is PTE# (0 to 1023)
1458 00005B66 C1E002             <1>     shl    eax, 2  ; shift 2 bits left to get PTE offset
1459 00005B69 01C3               <1>     add    ebx, eax ; now, EBX points to the relevant page table entry
1460                            <1> ; 17/04/2021 temporary
1461                            <1> ;   mov    eax, [ebx] ; get previous value of pte
1462                            <1> ;           ; bit 0 of EAX is always 0 (otherwise we would not be here)
1463                            <1>
1464                            <1> ; 17/04/2021
1465                            <1> ; ('swap_in' procedure call has been disabled as temporary)
1466                            <1> ;
1467                            <1> ;   and    eax, eax
1468                            <1> ;   jz     short pfh_gpte_1
1469                            <1> ;   ; 20/07/2015
1470                            <1> ;   xchg   ebx, ecx ; new page address (physical)
1471                            <1> ;   push   ebp ; 20/07/2015
1472                            <1> ;   mov    ebp, cr2
1473                            <1> ;           ; ECX = physical address of the page table entry
1474                            <1> ;           ; EBX = Memory page address (physical!)
1475                            <1> ;           ; EAX = Swap disk (offset) address
```

```
1476                            <1> ;           ; EBP = virtual address (page fault address)
1477                            <1> ;   call    swap_in
1478                            <1> ;   pop     ebp
1479                            <1> ;   jc      short pfh_err_retn
1480                            <1> ;   xchg    ecx, ebx
1481                            <1> ;           ; EBX = physical address of the page table entry
1482                            <1> ;           ; ECX = new page
1483                            <1> pfh_gpte_1:
1484 00005B6B 08D1             <1>     or      cl, dl  ; lower 3 bits are used as U/S, R/W, P flags
1485 00005B6D 890B             <1>     mov     [ebx], ecx ; Let's put the new page table entry here !
1486                            <1> pfh_cpp_ok:
1487                            <1> ; 17/04/2021
1488                            <1> ; ('add_to_swap_queue' procedure call has been disabled as temporary)
1489                            <1> ;
1490                            <1> ;   ; 20/07/2015
1491                            <1> ;   mov     ebx, cr2
1492                            <1> ;   call    add_to_swap_queue
1493                            <1>     ;
1494                            <1>     ; The new PTE (which contains the new page) will be added to
1495                            <1>     ; the swap queue, here.
1496                            <1>     ; (Later, if memory will become insufficient,
1497                            <1>     ; one page will be swapped out which is at the head of
1498                            <1>     ; the swap queue by using FIFO and access check methods.)
1499                            <1>     ;
1500 00005B6F 31C0             <1>     xor     eax, eax  ; 0
1501                            <1>     ;
1502                            <1> pfh_err_retn:
1503 00005B71 59               <1>     pop     ecx
1504 00005B72 5A               <1>     pop     edx
1505 00005B73 5B               <1>     pop     ebx
1506 00005B74 C3               <1>     retn
1507                            <1>
1508                            <1> copy_page:
1509                            <1>     ; 29/08/2023 (TRDOS 386 v2.0.6)
1510                            <1>     ; 22/09/2015
1511                            <1>     ; 21/09/2015
1512                            <1>     ; 19/09/2015
1513                            <1>     ; 07/09/2015
1514                            <1>     ; 31/08/2015
1515                            <1>     ; 20/07/2015
1516                            <1>     ; 05/05/2015
1517                            <1>     ; 03/05/2015
1518                            <1>     ; 18/04/2015
1519                            <1>     ; 12/04/2015
1520                            <1>     ; 30/10/2014
1521                            <1>     ; 18/10/2014 (Retro UNIX 386 v1 - beginning)
1522                            <1>     ;
1523                            <1>     ; INPUT ->
1524                            <1>     ;       EBX = Virtual (linear) address of source page
1525                            <1>     ;             (Page fault address)
1526                            <1>     ; OUTPUT ->
1527                            <1>     ;       EAX = PHYSICAL (real/flat) ADDRESS OF THE ALLOCATED PAGE
1528                            <1>     ;       (corresponding PAGE TABLE ENTRY is mapped/set)
1529                            <1>     ;       EAX = 0 (CF = 1)
1530                            <1>     ;               if there is not a free page to be allocated
1531                            <1>     ;       (page content of the source page will be copied
1532                            <1>     ;       onto the target/new page)
1533                            <1>     ;
1534                            <1>     ; Modified Registers -> ecx, ebx (except EAX)
1535                            <1>     ;
1536 00005B75 56               <1>     push    esi
1537 00005B76 57               <1>     push    edi
1538                            <1>     ;push   ebx
1539                            <1>     ;push   ecx
1540 00005B77 31F6             <1>     xor     esi, esi
1541 00005B79 C1EB0C           <1>     shr     ebx, 12 ; shift 12 bits right to get PDE & PTE numbers
1542 00005B7C 89D9             <1>     mov     ecx, ebx ; save page fault address (as 12 bit shifted)
1543 00005B7E C1EB08           <1>     shr     ebx, 8  ; shift 8 bits right and then
1544 00005B81 80E3FC           <1>     and     bl, 0FCh ; mask lower 2 bits to get PDE offset
1545 00005B84 89DF             <1>     mov     edi, ebx ; save it for the parent of current process
1546 00005B86 031D[8C8E0100]   <1>     add     ebx, [u.pgdir] ; EBX points to the relevant page dir entry
1547 00005B8C 8B03             <1>     mov     eax, [ebx] ; physical (base) address of the page table
1548 00005B8E 662500F0         <1>     and     ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
1549 00005B92 89CB             <1>     mov     ebx, ecx   ; (restore higher 20 bits of page fault address)
1550 00005B94 81E3FF030000     <1>     and     ebx, 3FFh  ; mask PDE# bits, the result is PTE# (0 to 1023)
1551                            <1>     ;shl    bx, 2      ; shift 2 bits left to get PTE offset
1552                            <1>     ; 29/08/2023
1553 00005B9A C1E302           <1>     shl     ebx, 2
1554 00005B9D 01C3             <1>     add     ebx, eax   ; EBX points to the relevant page table entry
1555                            <1>     ; 07/09/2015
1556 00005B9F 66F7030002       <1>         test    word [ebx], PTE_DUPLICATED ; (Does current process share this
1557                            <1>                                        ; read only page as a child process?)
1558 00005BA4 7509             <1>     jnz     short cpp_0 ; yes
1559 00005BA6 8B0B             <1>     mov     ecx, [ebx] ; PTE value
1560 00005BA8 6681E100F0       <1>     and     cx, PTE_A_CLEAR ; 0F000h  ; clear page attributes
1561 00005BAD EB31             <1>     jmp     short cpp_1
1562                            <1> cpp_0:
1563 00005BAF 89FE             <1>     mov     esi, edi
1564 00005BB1 0335[908E0100]   <1>     add     esi, [u.ppgdir] ; the parent's page directory entry
1565 00005BB7 8B06             <1>     mov     eax, [esi] ; physical (base) address of the page table
1566 00005BB9 662500F0         <1>     and     ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
1567 00005BBD 89CE             <1>     mov     esi, ecx   ; (restore higher 20 bits of page fault address)
1568 00005BBF 81E6FF030000     <1>     and     esi, 3FFh  ; mask PDE# bits, the result is PTE# (0 to 1023)
1569                            <1>     ;shl    si, 2      ; shift 2 bits left to get PTE offset
1570                            <1>     ; 29/08/2023
1571 00005BC5 C1E602           <1>     shl     esi, 2
1572 00005BC8 01C6             <1>     add     esi, eax   ; EDX points to the relevant page table entry
1573 00005BCA 8B0E             <1>     mov     ecx, [esi] ; PTE value of the parent process
1574                            <1>     ; 21/09/2015
1575 00005BCC 8B03             <1>     mov     eax, [ebx] ; PTE value of the child process
1576 00005BCE 662500F0         <1>     and     ax, PTE_A_CLEAR ; 0F000h ; clear page attributes
1577                            <1>     ;
1578 00005BD2 F6C101           <1>     test    cl, PTE_A_PRESENT ; is it a present/valid page ?
1579 00005BD5 7424             <1>     jz      short cpp_3 ; the parent's page is not same page
1580                            <1>     ;
1581 00005BD7 6681E100F0       <1>     and     cx, PTE_A_CLEAR ; 0F000h ; clear page attributes
1582 00005BDC 39C8             <1>     cmp     eax, ecx   ; Same page?
1583 00005BDE 751B             <1>     jne     short cpp_3 ; Parent page and child page are not same
1584                            <1>                         ; Convert child's page to writable page
1585                            <1> cpp_1:
1586 00005BE0 E841FCFFFF       <1>     call    allocate_page
1587 00005BE5 721A             <1>     jc      short cpp_4 ; 'insufficient memory' error
1588 00005BE7 21F6             <1>     and     esi, esi    ; check ESI is valid or not
1589 00005BE9 7405             <1>     jz      short cpp_2
1590                            <1>             ; Convert read only page to writable page
1591                            <1>             ;(for the parent of the current process)
1592                            <1>     ;and    word [esi], PTE_A_CLEAR ; 0F000h
1593                            <1>     ; 22/09/2015
1594 00005BEB 890E             <1>     mov     [esi], ecx
1595 00005BED 800E07           <1>     or      byte [esi], PTE_A_PRESENT + PTE_A_WRITE + PTE_A_USER
1596                            <1>                             ; 1+2+4 = 7
1597                            <1> cpp_2:
1598 00005BF0 89C7             <1>     mov     edi, eax ; new page address of the child process
1599                            <1>     ; 07/09/2015
```

```
1600 00005BF2 89CE              <1>     mov     esi, ecx ; the page address of the parent process
1601 00005BF4 B900040000        <1>     mov     ecx, PAGE_SIZE / 4
1602 00005BF9 F3A5              <1>     rep     movsd ; 31/08/2015
1603                            <1> cpp_3:
1604 00005BFB 0C07              <1>     or      al, PTE_A_PRESENT + PTE_A_WRITE + PTE_A_USER ; 1+2+4 = 7
1605 00005BFD 8903              <1>     mov     [ebx], eax ; Update PTE
1606 00005BFF 28C0              <1>     sub     al, al ; clear attributes
1607                            <1> cpp_4:
1608                            <1>     ;pop    ecx
1609                            <1>     ;pop    ebx
1610 00005C01 5F                <1>     pop     edi
1611 00005C02 5E                <1>     pop     esi
1612 00005C03 C3                <1>     retn
1613                            <1>
1614                            <1> ;; 28/04/2015
1615                            <1> ;; 24/10/2014
1616                            <1> ;; 21/10/2014 (Retro UNIX 386 v1 - beginning)
1617                            <1> ;; SWAP_PAGE_QUEUE (4096 bytes)
1618                            <1> ;;
1619                            <1> ;;   0000   0001   0002   0003   ....   1020   1021   1022   1023
1620                            <1> ;; +------+------+------+------+-    -+------+------+------+------+
1621                            <1> ;; |  pg1 |  pg2 |  pg3 |  pg4 | .... |pg1021|pg1022|pg1023|pg1024|
1622                            <1> ;; +------+------+------+------+-    -+------+------+------+------+
1623                            <1> ;;
1624                            <1> ;; [swpq_last] = 0 to 4096 (step 4) -> the last position on the queue
1625                            <1> ;;
1626                            <1> ;; Method:
1627                            <1> ;;  Swap page queue is a list of allocated pages with physical
1628                            <1> ;;  addresses (system mode virtual adresses = physical addresses).
1629                            <1> ;;  It is used for 'swap_in' and 'swap_out' procedures.
1630                            <1> ;;  When a new page is being allocated, swap queue is updated
1631                            <1> ;;  by 'swap_queue_shift' procedure, header of the queue (offset 0)
1632                            <1> ;;  is checked for 'accessed' flag. If the 1st page on the queue
1633                            <1> ;;  is 'accessed' or 'read only', it is dropped from the list;
1634                            <1> ;;  other pages from the 2nd to the last (in [swpq_last]) shifted
1635                            <1> ;;  to head then the 2nd page becomes the 1st and '[swpq_last]'
1636                            <1> ;;  offset value becomes it's previous offset value - 4.
1637                            <1> ;;  If the 1st page of the swap page queue is not 'accessed'
1638                            <1> ;;  the queue/list is not shifted.
1639                            <1> ;;  After the queue/list shift, newly allocated page is added
1640                            <1> ;;  to the tail of the queue at the [swpq_count*4] position.
1641                            <1> ;;  But, if [swpq_count] > 1023, the newly allocated page
1642                            <1> ;;  will not be added to the tail of swap page queue.
1643                            <1> ;;
1644                            <1> ;;  During 'swap_out' procedure, swap page queue is checked for
1645                            <1> ;;  the first non-accessed, writable page in the list,
1646                            <1> ;;  from the head to the tail. The list is shifted to left
1647                            <1> ;;  (to the head) till a non-accessed page will be found in the list.
1648                            <1> ;;  Then, this page       is swapped out (to disk) and then it is dropped
1649                            <1> ;;  from the list by a final swap queue shift. [swpq_count] value
1650                            <1> ;;  is changed. If all pages on the queue' are 'accessed',
1651                            <1> ;;  'insufficient memory' error will be returned ('swap_out'
1652                            <1> ;;  procedure will be failed)...
1653                            <1> ;;
1654                            <1> ;;  Note: If the 1st page of the queue is an 'accessed' page,
1655                            <1> ;;  'accessed' flag of the page will be reset (0) and that page
1656                            <1> ;;  (PTE) will be added to the tail of the queue after
1657                            <1> ;;  the check, if [swpq_count] < 1023. If [swpq_count] = 1024
1658                            <1> ;;  the queue will be rotated and the PTE in the head will be
1659                            <1> ;;  added to the tail after resetting 'accessed' bit.
1660                            <1> ;;
1661                            <1> ;;
1662                            <1> ;;
1663                            <1> ;; SWAP DISK/FILE (with 4096 bytes swapped page blocks)
1664                            <1> ;;
1665                            <1> ;;  00000000  00000004  00000008  0000000C   ...   size-8    size-4
1666                            <1> ;; +---------+---------+---------+---------+-- --+---------+---------+
1667                            <1> ;; |descriptr| page(1) | page(2) | page(3) | ... |page(n-1)| page(n) |
1668                            <1> ;; +---------+---------+---------+---------+-- --+---------+---------+
1669                            <1> ;;
1670                            <1> ;; [swpd_next] = the first free block address in swapped page records
1671                            <1> ;;                  for next free block search by 'swap_out' procedure.
1672                            <1> ;; [swpd_size] = swap disk/file size in sectors (512 bytes)
1673                            <1> ;;           NOTE: max. possible swap disk size is 1024 GB
1674                            <1> ;;           (entire swap space must be accessed by using
1675                            <1> ;;           31 bit offset address)
1676                            <1> ;; [swpd_free] = free block (4096 bytes) count in swap disk/file space
1677                            <1> ;; [swpd_start] = absolute/start address of the swap disk/file
1678                            <1> ;;            0 for file, or beginning sector of the swap partition
1679                            <1> ;; [swp_drv] = logical drive description table addr. of swap disk/file
1680                            <1> ;;
1681                            <1> ;;
1682                            <1> ;; Method:
1683                            <1> ;;  When the memory (ram) becomes insufficient, page allocation
1684                            <1> ;;  procedure swaps out a page from memory to the swap disk
1685                            <1> ;;  (partition) or swap file to get a new free page at the memory.
1686                            <1> ;;  Swapping out is performed by using swap page queue.
1687                            <1> ;;
1688                            <1> ;;  Allocation block size of swap disk/file is equal to page size
1689                            <1> ;;  (4096 bytes). Swapping address (in sectors) is recorded
1690                            <1> ;;  into relevant page file entry as 31 bit physical (logical)
1691                            <1> ;;  offset address as 1 bit shifted to left for present flag (0).
1692                            <1> ;;  Swapped page address is between 1 and swap disk/file size - 4.
1693                            <1> ;;  Absolute physical (logical) address of the swapped page is
1694                            <1> ;;  calculated by adding offset value to the swap partition's
1695                            <1> ;;  start address. If the swap device (disk) is a virtual disk
1696                            <1> ;;  or it is a file, start address of the swap disk/volume is 0,
1697                            <1> ;;  and offset value is equal to absolute (physical or logical)
1698                            <1> ;;  address/position. (It has not to be ZERO if the swap partition
1699                            <1> ;;  is in a partitioned virtual hard disk.)
1700                            <1> ;;
1701                            <1> ;;  Note: Swap addresses are always specified/declared in sectors,
1702                            <1> ;;  not in bytes or       in blocks/zones/clusters (4096 bytes) as unit.
1703                            <1> ;;
1704                            <1> ;;  Swap disk/file allocation is mapped via 'Swap Allocation Table'
1705                            <1> ;;  at memory as similar to 'Memory Allocation Table'.
1706                            <1> ;;
1707                            <1> ;;  Every bit of Swap Allocation Table repsesents one swap block
1708                            <1> ;;  (equal to page size) respectively. Bit 0 of the S.A.T. byte 0
1709                            <1> ;;  is reserved for swap disk/file block 0 as descriptor block
1710                            <1> ;;  (also for compatibility with PTE). If bit value is ZERO,
1711                            <1> ;;  it means relevant (respective) block is in use, and,
1712                            <1> ;;  of course, if bit value is 1, it means relevant (respective)
1713                            <1> ;;      swap disk/file block is free.
1714                            <1> ;;  For example: bit 1 of the byte 128 repsesents block 1025
1715                            <1> ;;  (128*8+1) or sector (offset) 8200 on the swap disk or
1716                            <1> ;;  byte (offset/position) 4198400 in the swap file.
1717                            <1> ;;  4GB swap space is represented via 128KB Swap Allocation Table.
1718                            <1> ;;  Initial layout of Swap Allocation Table is as follows:
1719                            <1> ;;  ------------------------------------------------------------
1720                            <1> ;;  01111111111111111111111111 .... 11111111111111111111111111111111
1721                            <1> ;;  ------------------------------------------------------------
1722                            <1> ;;  (0 is reserved block, 1s represent free blocks respectively.)
1723                            <1> ;;  (Note: Allocation cell/unit of the table is bit, not byte)
```

```
1724                    <1> ;;
1725                    <1> ;;  ..........................................................
1726                    <1> ;;
1727                    <1> ;;  'swap_out' procedure checks 'free_swap_blocks' count at first,
1728                    <1> ;;  then it searches Swap Allocation Table if free count is not
1729                    <1> ;;  zero. From begining the [swpd_next] dword value, the first bit
1730                    <1> ;;  position with value of 1 on the table is converted to swap
1731                    <1> ;;  disk/file offset address, in sectors (not 4096 bytes block).
1732                    <1> ;;  'ldrv_write' procedure is called with ldrv (logical drive
1733                    <1> ;;  number of physical swap disk or virtual swap disk)
1734                    <1> ;;  number, sector offset (not absolute sector -LBA- number),
1735                    <1> ;;  and sector count (8, 512*8 = 4096) and buffer adress
1736                    <1> ;;  (memory page). That will be a direct disk write procedure.
1737                    <1> ;;  (for preventing late memory allocation, significant waiting).
1738                    <1> ;;  If disk write procedure returns with error or free count of
1739                    <1> ;;  swap blocks is ZERO, 'swap_out' procedure will return with
1740                    <1> ;;  'insufficient memory error' (cf=1).
1741                    <1> ;;
1742                    <1> ;;  (Note: Even if free swap disk/file blocks was not zero,
1743                    <1> ;;  any disk write error will not be fixed by 'swap_out' procedure,
1744                    <1> ;;  in other words, 'swap_out' will not check the table for other
1745                    <1> ;;  free blocks after a disk write error. It will return to
1746                    <1> ;;  the caller with error (CF=1) which means swapping is failed.
1747                    <1> ;;
1748                    <1> ;;  After writing the page on to swap disk/file address/sector,
1749                    <1> ;;  'swap_out' procesure returns with that swap (offset) sector
1750                    <1> ;;  address (cf=0).
1751                    <1> ;;
1752                    <1> ;;  ..........................................................
1753                    <1> ;;
1754                    <1> ;;  'swap_in' procedure loads addressed (relevant) swap disk or
1755                    <1> ;;  file sectors at specified memory page. Then page allocation
1756                    <1> ;;  procedure updates relevant page table entry with 'present'
1757                    <1> ;;  attribute. If swap disk or file reading fails there is nothing
1758                    <1> ;;  to do, except to terminate the process which is the owner of
1759                    <1> ;;  the swapped page.
1760                    <1> ;;
1761                    <1> ;;  'swap_in' procedure sets the relevant/respective bit value
1762                    <1> ;;  in the Swap Allocation Table (as free block). 'swap_in' also
1763                    <1> ;;  updates [swpd_first] pointer if it is required.
1764                    <1> ;;
1765                    <1> ;;  ..........................................................
1766                    <1> ;;
1767                    <1> ;;  Note: If [swap_enabled] value is ZERO, that means there is not
1768                    <1> ;;  a swap disk or swap file in use... 'swap_in' and 'swap_out'
1769                    <1> ;;  procedures ans 'swap page que' procedures will not be active...
1770                    <1> ;;  'Insufficient memory' error will be returned by 'swap_out'
1771                    <1> ;;  and 'general protection fault' will be returned by 'swap_in'
1772                    <1> ;;  procedure, if it is called mistakenly (a wrong value in a PTE).
1773                    <1> ;;
1774                    <1>
1775                    <1> ; 17/04/2021
1776                    <1> ; ('swap_in' procedure call is disabled as temporary)
1777                    <1>
1778                    <1> swap_in:
1779                    <1>     ; 31/08/2015
1780                    <1>     ; 20/07/2015
1781                    <1>     ; 28/04/2015
1782                    <1>     ; 18/04/2015
1783                    <1>     ; 24/10/2014 (Retro UNIX 386 v1 - beginning)
1784                    <1>     ;
1785                    <1>     ; INPUT ->
1786                    <1>     ;       EBX = PHYSICAL (real/flat) ADDRESS OF THE MEMORY PAGE
1787                    <1>     ;       EBP = VIRTUAL (LINEAR) ADDRESS (page fault address)
1788                    <1>     ;       EAX = Offset Address for the swapped page on the
1789                    <1>     ;             swap disk or in the swap file.
1790                    <1>     ;
1791                    <1>     ; OUTPUT ->
1792                    <1>     ;       EAX = 0 if loading at memory has been successful
1793                    <1>     ;
1794                    <1>     ;       CF = 1 -> swap disk reading error (disk/file not present
1795                    <1>     ;                 or sector not present or drive not ready
1796                    <1>     ;            EAX = Error code
1797                    <1>     ;            [u.error] = EAX
1798                    <1>     ;                      = The last error code for the process
1799                    <1>     ;                        (will be reset after returning to user)
1800                    <1>     ;
1801                    <1>     ; Modified Registers -> EAX
1802                    <1>     ;
1803                    <1>
1804                    <1> ;       cmp     dword [swp_drv], 0
1805                    <1> ;   jna     short swpin_dnp_err
1806                    <1> ;
1807                    <1> ;   cmp     eax, [swpd_size]
1808                    <1> ;   jnb     short swpin_snp_err
1809                    <1> ;
1810                    <1> ;   push    esi
1811                    <1> ;   push    ebx
1812                    <1> ;   push    ecx
1813                    <1> ;   mov     esi, [swp_drv]
1814                    <1> ;   mov     ecx, PAGE_SIZE / LOGIC_SECT_SIZE  ; 8 !
1815                    <1> ;           ; Note: Even if corresponding physical disk's sector
1816                    <1> ;           ; size different than 512 bytes, logical disk sector
1817                    <1> ;           ; size is 512 bytes and disk reading procedure
1818                    <1> ;           ; will be performed for reading 4096 bytes
1819                    <1> ;           ; (2*2048, 8*512).
1820                    <1> ;   ; ESI = Logical disk description table address
1821                    <1> ;   ; EBX = Memory page (buffer) address (physical!)
1822                    <1> ;   ; EAX = Sector adress (offset address, logical sector number)
1823                    <1> ;   ; ECX = Sector count ; 8 sectors
1824                    <1> ;   push    eax
1825                    <1> ;   call    logical_disk_read
1826                    <1> ;   pop     eax
1827                    <1> ;   jnc     short swpin_read_ok
1828                    <1> ;   ;
1829                    <1> ;   mov     eax, SWP_DISK_READ_ERR ; drive not ready or read error
1830                    <1> ;   mov     [u.error], eax
1831                    <1> ;   jmp     short swpin_retn
1832                    <1> ;   ;
1833                    <1> ;swpin_read_ok:
1834                    <1> ;   ; EAX = Offset address (logical sector number)
1835                    <1> ;   call    unlink_swap_block  ; Deallocate swap block
1836                    <1> ;   ;
1837                    <1> ;   ; EBX = Memory page (buffer) address (physical!)
1838                    <1> ;   ; 20/07/2015
1839                    <1> ;   mov     ebx, ebp ; virtual address (page fault address)
1840                    <1> ;       and     bx, ~PAGE_OFF ; ~0FFFh ; reset bits, 0 to 11
1841                    <1> ;   mov     bl, [u.uno] ; current process number
1842                    <1> ;   ; EBX = Virtual (Linear) address & process number combination
1843                    <1> ;   call    swap_queue_shift
1844                    <1> ;   ; eax = 0 ; 10/06/2016 (if ebx input > 0, eax output = 0)
1845                    <1> ;   ;sub    eax, eax  ; 0 ; Error Code = 0  (no error)
1846                    <1> ;   ; zf = 1
1847                    <1> ;swpin_retn:
```

```
1848                    <1> ;   pop     ecx
1849                    <1> ;   pop     ebx
1850                    <1> ;   pop     esi
1851                    <1> ;   retn
1852                    <1> ;
1853                    <1> ;swpin_dnp_err:
1854                    <1> ;   mov     eax, SWP_DISK_NOT_PRESENT_ERR
1855                    <1> ;swpin_err_retn:
1856                    <1> ;   mov     [u.error], eax
1857                    <1> ;   stc
1858                    <1> ;   retn
1859                    <1> ;
1860                    <1> ;swpin_snp_err:
1861                    <1> ;   mov     eax, SWP_SECTOR_NOT_PRESENT_ERR
1862                    <1> ;   jmp     short swpin_err_retn
1863                    <1>
1864                    <1> ; 17/04/2021
1865                    <1> ; ('swap_out' procedure call is disabled as temporary)
1866                    <1>
1867                    <1> swap_out:
1868                    <1>     ; 10/06/2016
1869                    <1>     ; 07/06/2016
1870                    <1>       ; 23/05/2016
1871                    <1>     ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
1872                    <1>     ; 24/10/2014 - 31/08/2015 (Retro UNIX 386 v1)
1873                    <1>     ;
1874                    <1>     ; INPUT ->
1875                    <1>     ;       none
1876                    <1>     ;
1877                    <1>     ; OUTPUT ->
1878                    <1>     ;       EAX = Physical page address (which is swapped out
1879                    <1>     ;             for allocating a new page)
1880                    <1>     ;       CF = 1 -> swap disk writing error (disk/file not present
1881                    <1>     ;                 or sector not present or drive not ready
1882                    <1>     ;            EAX = Error code
1883                    <1>     ;            [u.error] = EAX
1884                    <1>     ;                      = The last error code for the process
1885                    <1>     ;                        (will be reset after returning to user)
1886                    <1>     ;
1887                    <1>     ; Modified Registers -> none (except EAX)
1888                    <1>     ;
1889                    <1>
1890                    <1> ;   cmp     word [swpq_count], 1
1891                    <1> ;       jc      swpout_im_err ; 'insufficient memory'
1892                    <1> ;
1893                    <1> ;       ;cmp    dword [swp_drv], 1
1894                    <1> ;   ;jc     short swpout_dnp_err ; 'swap disk/file not present'
1895                    <1> ;
1896                    <1> ;       cmp     dword [swpd_free], 1
1897                    <1> ;       jc      swpout_nfspc_err ; 'no free space on swap disk'
1898                    <1> ;
1899                    <1> ;   push    ebx ; *
1900                    <1> ;swpout_1:
1901                    <1> ;   ; 10/06/2016
1902                    <1> ;   xor     ebx, ebx ; shift the queue and return a PTE value
1903                    <1> ;   call    swap_queue_shift
1904                    <1> ;   and     eax, eax        ; 0 = empty queue (improper entries)
1905                    <1> ;       jz      swpout_npts_err         ; There is not any proper PTE
1906                    <1> ;                                       ; pointer in the swap queue
1907                    <1> ;   ; EAX = PTE value of the page
1908                    <1> ;   ; EBX = PTE address of the page
1909                    <1> ;   and     ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
1910                    <1> ;   ;
1911                    <1> ;   ; 07/06/2016
1912                    <1> ;   ; 19/05/2016
1913                    <1> ;   ; check this page is in timer events or not
1914                    <1> ;
1915                    <1> ;swpout_timer_page_0:
1916                    <1> ;   push    edx ; **
1917                    <1> ;
1918                    <1> ;   ; 07/06/2016
1919                    <1> ;   cmp     byte [timer_events], 0
1920                    <1> ;   jna     short swpout_2
1921                    <1> ;   ;
1922                    <1> ;   mov     dl, [timer_events]
1923                    <1> ;
1924                    <1> ;   push    ecx ; ***
1925                    <1> ;   push    ebx ; ****
1926                    <1> ;   mov     ebx, timer_set ; beginning address of timer event
1927                    <1> ;                          ; structures
1928                    <1> ;swpout_timer_page_1:
1929                    <1> ;   mov     cl, [ebx]
1930                    <1> ;   or      cl, cl ; 0 = free, >0 = process number
1931                    <1> ;   jz      short swpout_timer_page_3
1932                    <1> ;   mov     ecx, [ebx+12] ; response (signal return) address
1933                    <1> ;   and     cx, PTE_A_CLEAR ; clear offset part (right 12 bits)
1934                    <1> ;                           ; of the response byte address, to
1935                    <1> ;                           ; get beginning of the page address)
1936                    <1> ;   cmp     eax, ecx
1937                    <1> ;   jne     short swpout_timer_page_2 ; not same page
1938                    <1> ;
1939                    <1> ;   ; !same page!
1940                    <1> ;   ;
1941                    <1> ;   ; NOTE: // 19/05/2016 // - TRDOS 386 feature only ! -
1942                    <1> ;   ; This page will be used by the kernel to put timer event
1943                    <1> ;   ; response (signal return) byte at the requested address;
1944                    <1> ;   ; in order to prevent a possible wrong write (while
1945                    <1> ;   ; this page is swapped out) on physical memory,
1946                    <1> ;   ; we must protect this page against to be swapped out!
1947                    <1> ;   ;
1948                    <1> ;   pop     ebx ; ****
1949                    <1> ;   pop     ecx ; ***
1950                    <1> ;   pop     edx ; **
1951                    <1> ;   jmp     short swpout_1 ; do not swap out this page !
1952                    <1> ;
1953                    <1> ;swpout_timer_page_2:
1954                    <1> ;   ; 07/06/2016
1955                    <1> ;   dec     dl
1956                    <1> ;   jz      short swpout_timer_page_4
1957                    <1> ;swpout_timer_page_3:
1958                    <1> ;   ;cmp    ebx, timer_set + 240 ; last timer event (15*16)
1959                    <1> ;   ;jnb    short swpout_timer_page_4
1960                    <1> ;   add     ebx, 16
1961                    <1> ;   jmp     short swpout_timer_page_1
1962                    <1> ;
1963                    <1> ;swpout_timer_page_4:
1964                    <1> ;   pop     ebx ; ****
1965                    <1> ;   pop     ecx ; ***
1966                    <1> ;swpout_2:
1967                    <1> ;   mov     edx, ebx              ; Page table entry address
1968                    <1> ;   mov     ebx, eax              ; Buffer (Page) Address
1969                    <1> ;   ;
1970                    <1> ;   call    link_swap_block
1971                    <1> ;   jnc     short swpout_3        ; It may not be needed here
```

```
1972                    <1> ;                                   ; because [swpd_free] value
1973                    <1> ;                                   ; was checked at the beginging.
1974                    <1> ;   pop     edx ; **
1975                    <1> ;   pop     ebx ; *
1976                    <1> ;   jmp     short swpout_nfspc_err
1977                    <1> ;swpout_3:
1978                    <1> ;   test    eax, 80000000h ; test bit 31 (this may not be needed!)
1979                    <1> ;   jnz     short swpout_nfspc_err  ; 10/06/2016 (bit 31 = 1 !)
1980                    <1> ;   ;
1981                    <1> ;   push    esi ; **
1982                    <1> ;   push    ecx ; ***
1983                    <1> ;   push    eax ; sector address ; (31 bit !, bit 31 = 0)
1984                    <1> ;   mov     esi, [swp_drv]
1985                    <1> ;   mov     ecx, PAGE_SIZE / LOGIC_SECT_SIZE  ; 8 !
1986                    <1> ;           ; Note: Even if corresponding physical disk's sector
1987                    <1> ;           ; size different than 512 bytes, logical disk sector
1988                    <1> ;           ; size is 512 bytes and disk writing procedure
1989                    <1> ;           ; will be performed for writing 4096 bytes
1990                    <1> ;           ; (2*2048, 8*512).
1991                    <1> ;   ; ESI = Logical disk description table address
1992                    <1> ;   ; EBX = Buffer (Page) address
1993                    <1> ;   ; EAX = Sector adress (offset address, logical sector number)
1994                    <1> ;   ; ECX = Sector count ; 8 sectors
1995                    <1> ;   ; edx = PTE address
1996                    <1> ;   call    logical_disk_write
1997                    <1> ;   ; edx = PTE address
1998                    <1> ;   pop     ecx ; sector address
1999                    <1> ;   jnc     short swpout_write_ok
2000                    <1> ;   ;
2001                    <1> ;   ;; call unlink_swap_block ; this block must be left as 'in use'
2002                    <1> ;swpout_dw_err:
2003                    <1> ;   mov     eax, SWP_DISK_WRITE_ERR ; drive not ready or write error
2004                    <1> ;   mov     [u.error], eax
2005                    <1> ;   jmp     short swpout_retn
2006                    <1> ;   ;
2007                    <1> ;swpout_write_ok:
2008                    <1> ;   ; EBX = Buffer (page) address
2009                    <1> ;   ; EDX = Page Table Entry address
2010                    <1> ;   ; ECX = Swap disk sector (file block) address (31 bit)
2011                    <1> ;   shl     ecx, 1  ; 31 bit sector address from bit 1 to bit 31
2012                    <1> ;   mov     [edx], ecx
2013                    <1> ;           ; bit 0 = 0 (swapped page)
2014                    <1> ;   mov     eax, ebx
2015                    <1> ;swpout_retn:
2016                    <1> ;   pop     ecx ; ***
2017                    <1> ;   pop     esi ; **
2018                    <1> ;   pop     ebx ; *
2019                    <1> ;   retn
2020                    <1> ;
2021                    <1> ;;swpout_dnp_err:
2022                    <1> ;;  mov     eax, SWP_DISK_NOT_PRESENT_ERR ; disk not present
2023                    <1> ;;  jmp     short swpout_err_retn
2024                    <1> ;swpout_nfspc_err:
2025                    <1> ;   mov     eax, SWP_NO_FREE_SPACE_ERR ; no free space
2026                    <1> ;swpout_err_retn:
2027                    <1> ;   mov     [u.error], eax
2028                    <1> ;   ;stc
2029                    <1> ;   retn
2030                    <1> ;swpout_npts_err:
2031                    <1> ;   mov     eax, SWP_NO_PAGE_TO_SWAP_ERR
2032                    <1> ;   pop     ebx
2033                    <1> ;   jmp     short swpout_err_retn
2034                    <1> ;swpout_im_err:
2035                    <1> ;   mov     eax, ERR_MINOR_IM ; insufficient (out of) memory
2036                    <1> ;   jmp     short swpout_err_retn
2037                    <1>
2038                    <1> ; 17/04/2021
2039                    <1> ; ('swap_queue_shift' procedure call is disabled as temporary)
2040                    <1>
2041                    <1> swap_queue_shift:
2042                    <1>     ; 26/03/2017
2043                    <1>     ; 10/06/2016
2044                    <1>     ; 09/06/2016 - TRDOS 386 (TRDOS v2.0)
2045                    <1>     ; 23/10/2014 - 20/07/2015 (Retro UNIX 386 v1)
2046                    <1>     ;
2047                    <1>     ; INPUT ->
2048                    <1>     ;       EBX = Virtual (linear) address (bit 12 to 31)
2049                    <1>     ;             and process number combination (bit 0 to 11)
2050                    <1>     ;       EBX = 0 -> shift/drop from the head (offset 0)
2051                    <1>     ;
2052                    <1>     ; OUTPUT ->
2053                    <1>     ;       If EBX input > 0
2054                    <1>     ;          the queue will be shifted 4 bytes (dword),
2055                    <1>     ;          from the tail to the head, up to entry offset
2056                    <1>     ;          which points to EBX input value or nothing
2057                    <1>     ;          to do if EBX value is not found on the queue.
2058                    <1>     ;          (The entry -with EBX value- will be removed
2059                    <1>     ;          from the queue if it is found.)
2060                    <1>     ;
2061                    <1>     ;          EAX = 0
2062                    <1>     ;
2063                    <1>     ;       If EBX input = 0
2064                    <1>     ;          the queue will be shifted 4 bytes (dword),
2065                    <1>     ;          from the tail to the head, if the PTE address
2066                    <1>     ;          which is pointed in head of the queue is marked
2067                    <1>     ;          as "accessed" or it is marked as "non present".
2068                    <1>     ;          (If "accessed" flag of the PTE -which is pointed
2069                    <1>     ;          in the head- is set -to 1-, it will be reset
2070                    <1>     ;          -to 0- and then, the queue will be rotated
2071                    <1>     ;          -without dropping pointer of the PTE from
2072                    <1>     ;          the queue- for 4 bytes on head to tail direction.
2073                    <1>     ;          Pointer in the head will be moved into the tail,
2074                    <1>     ;          other PTEs will be shifted on head direction.)
2075                    <1>     ;
2076                    <1>     ;          Swap queue will be shifted up to the first
2077                    <1>     ;          'present' or 'non accessed' page will be found
2078                    <1>     ;          (as pointed) on the queue head (then it will be
2079                    <1>     ;           removed/dropped from the queue).
2080                    <1>     ;
2081                    <1>     ;          EAX (> 0) = PTE value of the page which is
2082                    <1>     ;               (it's pointer -virtual address-) dropped
2083                    <1>     ;               (removed) from swap queue.
2084                    <1>     ;          EBX = PTE address of the page (if EAX > 0)
2085                    <1>     ;                which is (it's pointer -virtual address-)
2086                    <1>     ;               dropped (removed) from swap queue.
2087                    <1>     ;
2088                    <1>     ;          EAX = 0 -> empty swap queue !
2089                    <1>     ;
2090                    <1>     ; Modified Registers -> EAX, EBX
2091                    <1>     ;
2092                    <1> ;   movzx   eax, word [swpq_count]  ; Max. 1024
2093                    <1> ;   and     ax, ax
2094                    <1> ;   jz      short swpqs_retn
2095                    <1> ;   push    edi
```

```
<1> ;   push    esi
<1> ;   push    ecx
<1> ;   mov     esi, swap_queue
<1> ;   mov     ecx, eax
<1> ;   or      ebx, ebx
<1> ;   jz      short swpqs_7
<1> ;swpqs_1:
<1> ;   lodsd
<1> ;   cmp     eax, ebx
<1> ;   je      short swpqs_2
<1> ;   loop    swpqs_1
<1> ;   ; 10/06/2016
<1> ;   sub     eax, eax
<1> ;   jmp     short swpqs_6
<1> ;swpqs_2:
<1> ;   mov     edi, esi
<1> ;   sub     edi, 4
<1> ;swpqs_3:
<1> ;   dec     word [swpq_count]
<1> ;   jz      short swpqs_5
<1> ;swpqs_4:
<1> ;   dec     ecx
<1> ;   rep     movsd   ; shift up (to the head)
<1> ;swpqs_5:
<1> ;   xor     eax, eax
<1> ;   mov     [edi], eax
<1> ;swpqs_6:
<1> ;   pop     ecx
<1> ;   pop     esi
<1> ;   pop     edi
<1> ;swpqs_retn:
<1> ;   retn
<1> ;swpqs_7:
<1> ;   mov     edi, esi ; head
<1> ;   lodsd
<1> ;   ; 20/07/2015
<1> ;   mov     ebx, eax
<1> ;   and     ebx, ~PAGE_OFF ; ~0FFFh
<1> ;                  ; ebx = virtual address (at page boundary)
<1> ;   and     eax, PAGE_OFF ; 0FFFh
<1> ;                  ; ax = process number (1 to 4095)
<1> ;   cmp     al, [u.uno]
<1> ;           ; Max. 16 (nproc) processes for Retro UNIX 386 v1
<1> ;   jne     short swpqs_8
<1> ;   mov     eax, [u.pgdir]
<1> ;   jmp     short swpqs_9
<1> ;swpqs_8:
<1> ;   ; 09/06/2016
<1> ;   cmp     byte [eax+p.stat-1], 0
<1> ;   jna     short swpqs_3     ; free (or terminated) process
<1> ;   cmp     byte [eax+p.stat-1], 2 ; waiting
<1> ;   ja      short swpqs_3    ; zombie (3) or undefined ?
<1> ;
<1> ;   ;shl    ax, 2
<1> ;   shl     al, 2
<1> ;   mov     eax, [eax+p.upage-4]
<1> ;   or      eax, eax
<1> ;   jz      short swpqs_3 ; invalid upage
<1> ;   add     eax, u.pgdir - user
<1> ;                  ; u.pgdir value for the process
<1> ;                  ; is in [eax]
<1> ;   mov     eax, [eax]
<1> ;   and     eax, eax
<1> ;   jz      short swpqs_3 ; invalid page directory
<1> ;swpqs_9:
<1> ;   push    edx
<1> ;   ; eax = page directory
<1> ;   ; ebx = virtual address
<1> ;   call    get_pte
<1> ;   mov     ebx, edx       ; PTE address
<1> ;   pop     edx
<1> ;   ; 10/06/2016
<1> ;   jc      short swpqs_13 ; empty PDE
<1> ;   ; EAX = PTE value
<1> ;   test    al, PTE_A_PRESENT ; bit 0 = 1
<1> ;   jz      short swpqs_13  ; Drop non-present page
<1> ;                          ; from the queue (head)
<1> ;   test    al, PTE_A_WRITE       ; bit 1 = 0 (read only)
<1> ;   jz      short swpqs_13  ; Drop read only page
<1> ;                          ; from the queue (head)
<1> ;   ;test   al, PTE_A_ACCESS ; bit 5 = 1 (Accessed)
<1> ;   ;jnz    short swpqs_11  ; present
<1> ;                          ; accessed page
<1> ;       btr     eax, PTE_A_ACCESS_BIT ; reset 'accessed' bit
<1> ;   jc      short swpqs_11  ; accessed page
<1> ;
<1> ;   dec     ecx
<1> ;   mov     [swpq_count], cx
<1> ;       jz      short swpqs_10
<1> ;           ; esi = head + 4
<1> ;           ; edi = head
<1> ;   rep     movsd   ; n = 1 to k-1, [n - 1] = [n]
<1> ;swpqs_10:
<1> ;   mov     [edi], ecx ; 0
<1> ;   jmp     short swpqs_6 ; 26/03/2017
<1> ;
<1> ;swpqs_11:
<1> ;   mov     [ebx], eax     ; save changed attribute
<1> ;   ; Rotation (head -> tail)
<1> ;   dec     ecx     ; entry count -> last entry number
<1> ;   jz      short swpqs_10
<1> ;           ; esi = head + 4
<1> ;           ; edi = head
<1> ;   mov     eax, [edi] ; 20/07/2015
<1> ;   rep     movsd   ; n = 1 to k-1, [n - 1] = [n]
<1> ;   mov     [edi], eax ; head -> tail ; [k] = [1]
<1> ;
<1> ;   mov     cx, [swpq_count]
<1> ;
<1> ;swpqs_12:
<1> ;   mov     esi, swap_queue ; head
<1> ;       jmp     swpqs_7
<1> ;
<1> ;swpqs_13:
<1> ;   dec     ecx
<1> ;   mov     [swpq_count], cx
<1> ;       jz      swpqs_5
<1> ;   jmp     short swpqs_12
<1>
<1> ; 17/04/2021
<1> ; ('add_to_swp_queue' procedure call is disabled as temporary)
<1>
<1> add_to_swap_queue:
<1>     ; 20/02/2017
```

```
2220                            <1>     ; 20/07/2015
2221                            <1>     ; 24/10/2014 (Retro UNIX 386 v1 - beginning)
2222                            <1>     ;
2223                            <1>     ; Adds new page to swap queue
2224                            <1>     ; (page directories and page tables must not be added
2225                            <1>     ; to swap queue)
2226                            <1>     ;
2227                            <1>     ; INPUT ->
2228                            <1>     ;       EBX = Linear (Virtual) addr for current process
2229                            <1>     ;       [u.uno]
2230                            <1>     ;       20/02/2017
2231                            <1>     ;       (Linear address = CORE + user's virtual address)
2232                            <1>     ;
2233                            <1>     ; OUTPUT ->
2234                            <1>     ;       EAX = [swpq_count]
2235                            <1>     ;             (after the PTE has been added)
2236                            <1>     ;       EAX = 0 -> Swap queue is full, (1024 entries)
2237                            <1>     ;             the PTE could not be added.
2238                            <1>     ;
2239                            <1>     ; Modified Registers -> EAX
2240                            <1>     ;
2241                            <1> ;   push    ebx
2242                            <1> ;       and     bx, ~PAGE_OFF ; ~0FFFh ; reset bits, 0 to 11
2243                            <1> ;   mov     bl, [u.uno] ; current process number
2244                            <1> ;   call    swap_queue_shift ; drop from the queue if
2245                            <1> ;                            ; it is already on the queue
2246                            <1> ;           ; then add it to the tail of the queue
2247                            <1> ;   movzx   eax, word [swpq_count]
2248                            <1> ;   cmp     ax, 1024
2249                            <1> ;   jb      short atsq_1
2250                            <1> ;   sub     ax, ax
2251                            <1> ;   pop     ebx
2252                            <1> ;   retn
2253                            <1> ;atsq_1:
2254                            <1> ;   push    esi
2255                            <1> ;   mov     esi, swap_queue
2256                            <1> ;   and     ax, ax
2257                            <1> ;   jz      short atsq_2
2258                            <1> ;   shl     ax, 2   ; convert to offset
2259                            <1> ;   add     esi, eax
2260                            <1> ;   shr     ax, 2
2261                            <1> ;atsq_2:
2262                            <1> ;   inc     ax
2263                            <1> ;   mov     [esi], ebx ; Virtual address + [u.uno] combination
2264                            <1> ;   mov     [swpq_count], ax
2265                            <1> ;   pop     esi
2266                            <1> ;   pop     ebx
2267                            <1> ;   retn
2268                            <1>
2269                            <1> ; 17/04/2021
2270                            <1> ; ('unlink_swap_block' procedure call is disabled as temporary)
2271                            <1>
2272                            <1> unlink_swap_block:
2273                            <1>     ; 15/09/2015
2274                            <1>     ; 30/04/2015
2275                            <1>     ; 18/04/2015
2276                            <1>     ; 24/10/2014 (Retro UNIX 386 v1 - beginning)
2277                            <1>     ;
2278                            <1>     ; INPUT ->
2279                            <1>     ;       EAX = swap disk/file offset address
2280                            <1>     ;             (bit 1 to bit 31)
2281                            <1>     ; OUTPUT ->
2282                            <1>     ;       [swpd_free] is increased
2283                            <1>     ;       (corresponding SWAP DISK ALLOC. TABLE bit is SET)
2284                            <1>     ;
2285                            <1>     ; Modified Registers -> EAX
2286                            <1>     ;
2287                            <1> ;   push    ebx
2288                            <1> ;   push    edx
2289                            <1> ;   ;
2290                            <1> ;   shr     eax, SECTOR_SHIFT+1  ;3+1 ; shift sector address to
2291                            <1> ;                                ; 3 bits right
2292                            <1> ;                                ; to get swap block/page number
2293                            <1> ;   mov     edx, eax
2294                            <1> ;   ; 15/09/2015
2295                            <1> ;   shr     edx, 3               ; to get offset to S.A.T.
2296                            <1> ;                                ; (1 allocation bit = 1 page)
2297                            <1> ;                                ; (1 allocation bytes = 8 pages)
2298                            <1> ;   and     dl, 0FCh             ; clear lower 2 bits
2299                            <1> ;                                ; (to get 32 bit position)
2300                            <1> ;   ;
2301                            <1> ;   mov     ebx, swap_alloc_table ; Swap Allocation Table address
2302                            <1> ;   add     ebx, edx
2303                            <1> ;   and     eax, 1Fh             ; lower 5 bits only
2304                            <1> ;                                ; (allocation bit position)
2305                            <1> ;   cmp     eax, [swpd_next]     ; is the new free block addr. lower
2306                            <1> ;                                ; than the address in 'swpd_next' ?
2307                            <1> ;                                ; (next/first free block value)
2308                            <1> ;   jnb     short uswpbl_1       ; no
2309                            <1> ;   mov     [swpd_next], eax      ; yes
2310                            <1> ;uswpbl_1:
2311                            <1> ;   bts     [ebx], eax           ; unlink/release/deallocate block
2312                            <1> ;                                ; set relevant bit to 1.
2313                            <1> ;                                ; set CF to the previous bit value
2314                            <1> ;   cmc                          ; complement carry flag
2315                            <1> ;   jc      short uswpbl_2       ; do not increase swfd_free count
2316                            <1> ;                                ; if the block is already deallocated
2317                            <1> ;                                ; before.
2318                            <1> ;       inc     dword [swpd_free]
2319                            <1> ;uswpbl_2:
2320                            <1> ;   pop     edx
2321                            <1> ;   pop     ebx
2322                            <1> ;   retn
2323                            <1>
2324                            <1> ; 17/04/2021
2325                            <1> ; ('ink_swap_block' procedure call is disabled as temporary)
2326                            <1>
2327                            <1> link_swap_block:
2328                            <1>     ; 01/07/2015
2329                            <1>     ; 18/04/2015
2330                            <1>     ; 24/10/2014 (Retro UNIX 386 v1 - beginning)
2331                            <1>     ;
2332                            <1>     ; INPUT -> none
2333                            <1>     ;
2334                            <1>     ; OUTPUT ->
2335                            <1>     ;       EAX = OFFSET ADDRESS OF THE ALLOCATED BLOCK (4096 bytes)
2336                            <1>     ;             in sectors (corresponding
2337                            <1>     ;             SWAP DISK ALLOCATION TABLE bit is RESET)
2338                            <1>     ;
2339                            <1>     ;       CF = 1 and EAX = 0
2340                            <1>     ;                  if there is not a free block to be allocated
2341                            <1>     ;
2342                            <1>     ; Modified Registers -> none (except EAX)
2343                            <1>     ;
```

```
2344                            <1>
2345                            <1> ;   ;mov    eax, [swpd_free]
2346                            <1> ;   ;and    eax, eax
2347                            <1> ;   ;jz     short out_of_swpspc
2348                            <1> ;   ;
2349                            <1> ;   push    ebx
2350                            <1> ;   push    ecx
2351                            <1> ;   ;
2352                            <1> ;   mov     ebx, swap_alloc_table ; Swap Allocation Table offset
2353                            <1> ;   mov     ecx, ebx
2354                            <1> ;   add     ebx, [swpd_next] ; Free block searching starts from here
2355                            <1> ;                            ; next_free_swap_block >> 5
2356                            <1> ;   add     ecx, [swpd_last] ; Free block searching ends here
2357                            <1> ;                            ; (total_swap_blocks - 1) >> 5
2358                            <1> ;lswbl_scan:
2359                            <1> ;   cmp     ebx, ecx
2360                            <1> ;   ja      short lswbl_notfound
2361                            <1> ;   ;
2362                            <1> ;   bsf     eax, [ebx] ; Scans source operand for first bit set (1).
2363                            <1> ;                      ; Clears ZF if a bit is found set (1) and
2364                            <1> ;                      ; loads the destination with an index to
2365                            <1> ;                      ; first set bit. (0 -> 31)
2366                            <1> ;                      ; Sets ZF to 1 if no bits are found set.
2367                            <1> ;   ; 01/07/2015
2368                            <1> ;   jnz     short lswbl_found ; ZF = 0 -> a free block has been found
2369                            <1> ;                     ;
2370                            <1> ;                     ; NOTE:  a Swap Disk Allocation Table bit
2371                            <1> ;                     ;       with value of 1 means
2372                            <1> ;                     ;       the corresponding page is free
2373                            <1> ;                     ;       (Retro UNIX 386 v1 feaure only!)
2374                            <1> ;   add     ebx, 4
2375                            <1> ;                     ; We return back for searching next page block
2376                            <1> ;                     ; NOTE: [swpd_free] is not ZERO; so,
2377                            <1> ;                     ;       we always will find at least 1 free block here.
2378                            <1> ;   jmp     short lswbl_scan
2379                            <1> ;   ;
2380                            <1> ;lswbl_notfound:
2381                            <1> ;   sub     ecx, swap_alloc_table
2382                            <1> ;   mov     [swpd_next], ecx ; next/first free page = last page
2383                            <1> ;                            ; (unlink_swap_block procedure will change it)
2384                            <1> ;   xor     eax, eax
2385                            <1> ;   mov     [swpd_free], eax
2386                            <1> ;   stc
2387                            <1> ;lswbl_ok:
2388                            <1> ;   pop     ecx
2389                            <1> ;   pop     ebx
2390                            <1> ;   retn
2391                            <1> ;   ;
2392                            <1> ;;out_of_swpspc:
2393                            <1> ;;  stc
2394                            <1> ;;  retn
2395                            <1> ;
2396                            <1> ;lswbl_found:
2397                            <1> ;   mov     ecx, ebx
2398                            <1> ;   sub     ecx, swap_alloc_table
2399                            <1> ;   mov     [swpd_next], ecx ; Set first free block searching start
2400                            <1> ;                            ; address/offset (to the next)
2401                            <1> ;       dec     dword [swpd_free] ; 1 block has been allocated (X = X-1)
2402                            <1> ;   ;
2403                            <1> ;   btr     [ebx], eax      ; The destination bit indexed by the source value
2404                            <1> ;                           ; is copied into the Carry Flag and then cleared
2405                            <1> ;                           ; in the destination.
2406                            <1> ;                           ;
2407                            <1> ;                           ; Reset the bit which is corresponding to the
2408                            <1> ;                           ; (just) allocated block.
2409                            <1> ;   shl     ecx, 5          ; (block offset * 32) + block index
2410                            <1> ;   add     eax, ecx        ; = block number
2411                            <1> ;   shl     eax, SECTOR_SHIFT ; 3, sector (offset) address of the block
2412                            <1> ;                           ; 1 block =  8 sectors
2413                            <1> ;   ;
2414                            <1> ;   ; EAX = offset address of swap disk/file sector (beginning of the block)
2415                            <1> ;   ;
2416                            <1> ;   ; NOTE: The relevant page table entry will be updated
2417                            <1> ;   ;       according to this EAX value...
2418                            <1> ;   ;
2419                            <1> ;   jmp     short lswbl_ok
2420                            <1>
2421                            <1> ; 17/04/2021
2422                            <1> ; ('logical_disk_read' procedure call is disabled as temporary)
2423                            <1>
2424                            <1> logical_disk_read:
2425                            <1>     ; 20/07/2015
2426                            <1>     ; 09/03/2015 (temporary code here)
2427                            <1>     ;
2428                            <1>     ; INPUT ->
2429                            <1>     ;       ESI = Logical disk description table address
2430                            <1>     ;       EBX = Memory page (buffer) address (physical!)
2431                            <1>     ;       EAX = Sector adress (offset address, logical sector number)
2432                            <1>     ;       ECX = Sector count
2433                            <1>     ;
2434                            <1>     ;
2435                            <1> ;   retn
2436                            <1>
2437                            <1> ; 17/04/2021
2438                            <1> ; ('logical_disk_write' procedure call is disabled as temporary)
2439                            <1>
2440                            <1> logical_disk_write:
2441                            <1>     ; 20/07/2015
2442                            <1>     ; 09/03/2015 (temporary code here)
2443                            <1>     ;
2444                            <1>     ; INPUT ->
2445                            <1>     ;       ESI = Logical disk description table address
2446                            <1>     ;       EBX = Memory page (buffer) address (physical!)
2447                            <1>     ;       EAX = Sector adress (offset address, logical sector number)
2448                            <1>     ;       ECX = Sector count
2449                            <1>     ;
2450                            <1> ;   retn
2451                            <1>
2452                            <1> get_physical_addr:
2453                            <1>     ; 17/04/2021 - TRDOS 386 v2.0.4
2454                            <1>     ;       (temporary modifications)
2455                            <1>     ;
2456                            <1>     ; 26/03/2017
2457                            <1>     ; 20/02/2017
2458                            <1>     ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
2459                            <1>     ; 18/10/2015
2460                            <1>     ; 29/07/2015
2461                            <1>     ; 20/07/2015
2462                            <1>     ; 04/06/2015
2463                            <1>     ; 20/05/2015
2464                            <1>     ; 28/04/2015
2465                            <1>     ; 18/04/2015
2466                            <1>     ; Get physical address
2467                            <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>     ;
00005C04 81C300004000           <1>     add     ebx, CORE ; 18/10/2015
                                <1> get_physical_addr_x: ; 27/05/2016
00005C0A A1[8C8E0100]           <1>     mov     eax, [u.pgdir]
00005C0F E824FDFFFF             <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 ?
00005C14 731C                   <1>     jnc     short gpa_1
                                <1>     ;
00005C16 E80BFCFFFF             <1>     call    allocate_page
00005C1B 723F                   <1>     jc      short gpa_im_err  ; 'insufficient memory' error
                                <1> gpa_0:
00005C1D E875FCFFFF             <1>     call    clear_page
                                <1>     ; EAX = Physical (base) address of the allocated (new) page
00005C22 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)
00005C24 8902                   <1>     mov     [edx], eax ; Let's put the new page directory entry here !
00005C26 A1[8C8E0100]           <1>     mov     eax, [u.pgdir]
00005C2B E808FDFFFF             <1>     call    get_pte
00005C30 722A                   <1>     jc      short gpa_im_err ; 'insufficient memory' error
                                <1> gpa_1:
                                <1>     ; EAX = PTE value, EDX = PTE address
00005C32 A801                   <1>     test    al, PTE_A_PRESENT
00005C34 7516                   <1>     jnz     short gpa_3 ; 26/03/2017
00005C36 09C0                   <1>     or      eax, eax
00005C38 7446                   <1>     jz      short gpa_7  ; Allocate a new page
                                <1>
                                <1> ; 17/04/2021 (TRDOS 386 v2.0.4)
                                <1> ; ('reload_page' procedure call is disabled as temporary)
00005C3A EB20                   <1>     jmp     short gpa_im_err  ; temporary !
                                <1>
                                <1>     ; 20/07/2015
                                <1> ;   push    ebp
                                <1> ;   mov     ebp, ebx ; virtual (linear) address
                                <1> ;   ; reload swapped page
                                <1> ;   call    reload_page ; 28/04/2015
                                <1> ;   pop     ebp
                                <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>     ;
00005C3C 803D[1E880100]00       <1>     cmp     byte [no_page_swap], 0
00005C43 761D                   <1>     jna     short gpa_4 ; this page can be swapped out
                                <1>     ; this page must not be swapped out
                                <1>     ; but 'no_page_swap' must be reset here
                                <1>     ; immediately for other callers (*)
                                <1>     ; (otherwise, swap queue would not be long enough)
00005C45 E844000000             <1>     call    gpa_8 ; 26/03/2017
00005C4A EB16                   <1>     jmp     short gpa_5
                                <1> gpa_3:
                                <1>     ; 26/03/2017
00005C4C 803D[1E880100]00       <1>     cmp     byte [no_page_swap], 0
00005C53 7611                   <1>     jna     short gpa_6 ; this page can be swapped out
00005C55 E834000000             <1>     call    gpa_8
00005C5A EB0A                   <1>     jmp     short gpa_6
                                <1>
                                <1> gpa_im_err:
00005C5C B804000000             <1>     mov     eax, ERR_MINOR_IM ; Insufficient memory (minor) error!
                                <1>                               ; Major error = 0 (No protection fault)
00005C61 C3                     <1>     retn
                                <1> gpa_4:
                                <1> ; 17/04/2021 (TRDOS 386 v2.0.4)
                                <1> ; ('add_to_swap_queue' procedure call is disabled as temporary)
                                <1>
                                <1>     ; 20/07/2015
                                <1>     ; 20/05/2015
                                <1>     ; add this page to swap queue
                                <1> ;   push    eax
                                <1> ;   ; EBX = Linear (CORE+virtual) address ; 20/02/2017
                                <1> ;   call    add_to_swap_queue
                                <1> ;   pop     eax
                                <1> gpa_5:
                                <1>             ; PTE address in EDX
                                <1>             ; virtual address in EBX
                                <1>     ; EAX = memory page address
00005C62 0C07                   <1>     or      al, PTE_A_PRESENT + PTE_A_USER + PTE_A_WRITE
                                <1>                               ; present flag, bit 0 = 1
                                <1>                               ; user flag, bit 2 = 1
                                <1>                               ; writable flag, bit 1 = 1
00005C64 8902                   <1>     mov     [edx], eax  ; Update PTE value
                                <1> gpa_6:
                                <1>     ; 18/10/2015
00005C66 89D9                   <1>     mov     ecx, ebx
00005C68 81E1FF0F0000           <1>     and     ecx, PAGE_OFF
00005C6E 89C2                   <1>     mov     edx, eax
00005C70 662500F0               <1>     and     ax, PTE_A_CLEAR
00005C74 01C8                   <1>     add     eax, ecx
00005C76 F7D9                   <1>     neg     ecx ; 1 -> -1 (0FFFFFFFFh), 4095 (0FFFh) -> -4095
00005C78 81C100100000           <1>     add     ecx, PAGE_SIZE
00005C7E F8                     <1>     clc
                                <1> gpa_retn:
00005C7F C3                     <1>     retn
```

```
                                <1> gpa_7:
00005C80 E8A1FBFFFF             <1>     call    allocate_page
00005C85 72D5                   <1>     jc      short gpa_im_err ; 'insufficient memory' error
00005C87 E80BFCFFFF             <1>     call    clear_page
00005C8C EBAE                   <1>     jmp     short gpa_2
                                <1>
                                <1> gpa_8: ; 26/03/2017
00005C8E C605[1E880100]00       <1>     mov     byte [no_page_swap], 0
                                <1>
00005C95 C3                     <1>     retn    ; 17/04/2021 (temporary)
                                <1>
                                <1> ; 17/04/2021 (TRDOS 386 v2.0.4)
                                <1> ; ('swap_queue_shift' procedure call is disabled as temporary)
                                <1> ;
                                <1> ;   push    ebx
                                <1> ;   push    eax ; 26/03/2017
                                <1> ;       and     bx, ~PAGE_OFF ; ~0FFFh ; reset bits, 0 to 11
                                <1> ;   mov     bl, [u.uno] ; current process number
                                <1> ;   call    swap_queue_shift ; drop from the queue if
                                <1> ;                            ; it is already on the queue
                                <1> ;   pop     eax ; 26/03/2017
                                <1> ;   pop     ebx
                                <1> ;
                                <1> ;   retn
                                <1>
                                <1> ; 17/04/2021
                                <1> ; ('reload_page' procedure call is disabled as temporary)
                                <1>
                                <1> reload_page:
                                <1>     ; 20/07/2015
                                <1>     ; 28/04/2015 (Retro UNIX 386 v1 - beginning)
                                <1>     ;
                                <1>     ; Reload (Restore) swapped page at memory
                                <1>     ;
                                <1>     ; INPUT ->
                                <1>     ;       EBP = Virtual (linear) memory address
                                <1>     ;       EAX = PTE value (swap disk sector address)
                                <1>     ;       (Swap disk sector address = bit 1 to bit 31 of EAX)
                                <1>     ; OUTPUT ->
                                <1>     ;       EAX = PHYSICAL (real/flat) ADDRESS OF RELOADED PAGE
                                <1>     ;
                                <1>     ;       CF = 1 and EAX = error code
                                <1>     ;
                                <1>     ; Modified Registers -> none (except EAX)
                                <1>     ;
                                <1> ;   shr     eax, 1   ; Convert PTE value to swap disk address
                                <1> ;   push    ebx      ;
                                <1> ;   mov     ebx, eax ; Swap disk (offset) address
                                <1> ;   call    allocate_page
                                <1> ;   jc      short rlp_im_err
                                <1> ;   xchg    eax, ebx
                                <1> ;   ; EBX = Physical memory (page) address
                                <1> ;   ; EAX = Swap disk (offset) address
                                <1> ;   ; EBP = Virtual (linear) memory address
                                <1> ;   call    swap_in
                                <1> ;   jc      short rlp_swp_err  ; (swap disk/file read error)
                                <1> ;   mov     eax, ebx
                                <1> ;rlp_retn:
                                <1> ;   pop     ebx
                                <1> ;   retn
                                <1> ;
                                <1> ;rlp_im_err:
                                <1> ;   mov     eax, ERR_MINOR_IM ; Insufficient memory (minor) error!
                                <1> ;                             ; Major error = 0 (No protection fault)
                                <1> ;   jmp     short rlp_retn
                                <1> ;
                                <1> ;rlp_swp_err:
                                <1> ;   mov     eax, SWP_DISK_READ_ERR ; Swap disk read error !
                                <1> ;   jmp     short rlp_retn
                                <1>
                                <1> copy_page_dir:
                                <1>     ; 17/04/2021 (temporary modifications)
                                <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>     ;
00005C96 E88BFBFFFF             <1>     call    allocate_page
00005C9B 723E                   <1>     jc      short cpd_err
                                <1>     ;
00005C9D 55                     <1>     push    ebp ; 20/07/2015
00005C9E 56                     <1>     push    esi
00005C9F 57                     <1>     push    edi
00005CA0 53                     <1>     push    ebx
00005CA1 51                     <1>     push    ecx
00005CA2 8B35[8C8E0100]         <1>     mov     esi, [u.pgdir]
00005CA8 89C7                   <1>     mov     edi, eax
00005CAA 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)
00005CAB A5                     <1>     movsd
00005CAC BD00004000             <1>     mov     ebp, 1024*4096 ; pass the 1st 4MB (system space)
00005CB1 B9FF030000             <1>     mov     ecx, (PAGE_SIZE / 4) - 1 ; 1023
                                <1> cpd_0:
00005CB6 AD                     <1>     lodsd
                                <1>     ;or     eax, eax
                                <1>         ;jnz     short cpd_1
00005CB7 A801                   <1>     test    al, PDE_A_PRESENT ;  bit 0 =  1
00005CB9 7508                   <1>     jnz     short cpd_1
                                <1>     ; (virtual address at the end of the page table)
00005CBB 81C500004000           <1>     add     ebp, 1024*4096 ; page size * PTE count
00005CC1 EB0F                   <1>     jmp     short cpd_2
                                <1> cpd_1:
00005CC3 662500F0               <1>     and     ax, PDE_A_CLEAR ; 0F000h ; clear attribute bits
00005CC7 89C3                   <1>     mov     ebx, eax
                                <1>     ; EBX = Parent's page table address
00005CC9 E81F000000             <1>     call    copy_page_table
00005CCE 720C                   <1>     jc      short cpd_p_err
                                <1>     ; EAX = Child's page table address
00005CD0 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)
```

```
2716                                    <1> cpd_2:
2717 00005CD2 AB                       <1>     stosd
2718 00005CD3 E2E1                     <1>     loop    cpd_0
2719                                    <1>     ;
2720 00005CD5 58                       <1>     pop     eax  ; restore child's page directory address
2721                                    <1> cpd_3:
2722 00005CD6 59                       <1>     pop     ecx
2723 00005CD7 5B                       <1>     pop     ebx
2724 00005CD8 5F                       <1>     pop     edi
2725 00005CD9 5E                       <1>     pop     esi
2726 00005CDA 5D                       <1>     pop     ebp
2727                                    <1> cpd_err:
2728 00005CDB C3                       <1>     retn
2729                                    <1> cpd_p_err:
2730                                    <1>     ; release the allocated pages missing (recover free space)
2731 00005CDC 58                       <1>     pop     eax  ; the new page directory address (physical)
2732 00005CDD 8B1D[8C8E0100]           <1>     mov     ebx, [u.pgdir] ; parent's page directory address
2733 00005CE3 E86EFCFFFF               <1>     call    deallocate_page_dir
2734 00005CE8 29C0                     <1>     sub     eax, eax ; 0
2735 00005CEA F9                       <1>     stc
2736 00005CEB EBE9                     <1>     jmp     short cpd_3
2737                                    <1>
2738                                    <1> copy_page_table:
2739                                    <1>     ; 17/04/2021 (temporary modifications)
2740                                    <1>     ; 19/09/2015
2741                                    <1>     ; temporary - 07/09/2015
2742                                    <1>     ; 07/09/2015 (Retro UNIX 386 v1 - beginning)
2743                                    <1>     ;
2744                                    <1>     ; INPUT ->
2745                                    <1>     ;       EBX = PHYSICAL (real/flat) ADDRESS of the parent's page table.
2746                                    <1>     ;       EBP = page table entry index (from 'copy_page_dir')
2747                                    <1>     ; OUTPUT ->
2748                                    <1>     ;       EAX = PHYSICAL (real/flat) ADDRESS of the child's page table.
2749                                    <1>     ;       EBP = (recent) page table index (for 'add_to_swap_queue')
2750                                    <1>     ;       CF = 1 -> error
2751                                    <1>     ;
2752                                    <1>     ; Modified Registers -> EBP (except EAX)
2753                                    <1>     ;
2754 00005CED E834FBFFFF               <1>     call    allocate_page
2755 00005CF2 7244                     <1>     jc      short cpt_err
2756                                    <1>     ;
2757 00005CF4 50                       <1>     push    eax ; *
2758                                    <1>     ;push   ebx
2759 00005CF5 56                       <1>     push    esi
2760 00005CF6 57                       <1>     push    edi
2761 00005CF7 52                       <1>     push    edx
2762 00005CF8 51                       <1>     push    ecx
2763                                    <1>     ;
2764 00005CF9 89DE                     <1>     mov     esi, ebx
2765 00005CFB 89C7                     <1>     mov     edi, eax
2766 00005CFD 89C2                     <1>     mov     edx, eax
2767 00005CFF 81C200100000             <1>     add     edx, PAGE_SIZE
2768                                    <1> cpt_0:
2769 00005D05 AD                       <1>     lodsd
2770 00005D06 A801                     <1>     test    al, PTE_A_PRESENT ;  bit 0 = 1
2771                                    <1>     ;jnz    short cpt_1 (*)
2772                                    <1>     ; 17/04/2021 (temporary (*)
2773                                    <1>     ;and    eax, eax (*)
2774 00005D08 741E                     <1>     jz      short cpt_2  ; 17/04/2021
2775                                    <1>
2776                                    <1> ; 17/04/2021
2777                                    <1> ; ('reload_page' procedure call is disabled as temporary)
2778                                    <1> ;
2779                                    <1> ;   ; ebp = virtual (linear) address of the memory page
2780                                    <1> ;   call    reload_page ; 28/04/2015
2781                                    <1> ;   jc      short cpt_p_err
2782                                    <1> cpt_1:
2783 00005D0A 662500F0                 <1>     and     ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
2784 00005D0E 89C1                     <1>     mov     ecx, eax
2785                                    <1>     ; Allocate a new page for the child process
2786 00005D10 E811FBFFFF               <1>     call    allocate_page
2787 00005D15 721C                     <1>     jc      short cpt_p_err
2788 00005D17 57                       <1>     push    edi
2789 00005D18 56                       <1>     push    esi
2790 00005D19 89CE                     <1>     mov     esi, ecx
2791 00005D1B 89C7                     <1>     mov     edi, eax
2792 00005D1D B900040000               <1>     mov     ecx, PAGE_SIZE/4
2793 00005D22 F3A5                     <1>     rep     movsd   ; copy page (4096 bytes)
2794 00005D24 5E                       <1>     pop     esi
2795 00005D25 5F                       <1>     pop     edi
2796                                    <1>     ;
2797                                    <1>
2798                                    <1> ; 17/04/2021
2799                                    <1> ; ('add_to_swap_queue' procedure call is disabled as temporary)
2800                                    <1> ;
2801                                    <1> ;   push    ebx
2802                                    <1> ;   push    eax
2803                                    <1> ;   mov     ebx, ebp
2804                                    <1> ;   ; ebx = virtual address of the memory page
2805                                    <1> ;   call    add_to_swap_queue
2806                                    <1> ;   pop     eax
2807                                    <1> ;   pop     ebx
2808                                    <1>     ;
2809                                    <1>     ;or     ax, PTE_A_USER+PTE_A_PRESENT
2810 00005D26 0C07                     <1>     or      al, PTE_A_USER+PTE_A_WRITE+PTE_A_PRESENT
2811                                    <1> cpt_2:
2812 00005D28 AB                       <1>     stosd  ; EDI points to child's PTE
2813                                    <1>     ;
2814 00005D29 81C500100000             <1>     add     ebp, 4096 ; 20/07/2015 (next page)
2815                                    <1>     ;
2816 00005D2F 39D7                     <1>     cmp     edi, edx
2817 00005D31 72D2                     <1>     jb      short cpt_0
2818                                    <1> cpt_p_err:
2819 00005D33 59                       <1>     pop     ecx
2820 00005D34 5A                       <1>     pop     edx
2821 00005D35 5F                       <1>     pop     edi
2822 00005D36 5E                       <1>     pop     esi
2823                                    <1>     ;pop    ebx
2824 00005D37 58                       <1>     pop     eax ; *
2825                                    <1> cpt_err:
2826 00005D38 C3                       <1>     retn
2827                                    <1>
2828                                    <1> allocate_memory_block:
2829                                    <1>     ; 01/05/2017
2830                                    <1>     ; 28/04/2017
2831                                    <1>     ; 25/04/2017
2832                                    <1>     ; 01/04/2016, 02/04/2016, 03/04/2016
2833                                    <1>     ; 13/03/2016, 14/03/2016
2834                                    <1>     ; 12/03/2016 (TRDOS 386 = TRDOS v2.0)
2835                                    <1>     ; Allocating contiguous memory pages (in the kernel's memory space)
2836                                    <1>     ;
2837                                    <1>     ; INPUT ->
2838                                    <1>     ;       EAX = Beginning address (physical)
2839                                    <1>     ;       EAX = 0 -> Allocate memory block from the first proper aperture
```

```
                                <1>      ;       ECX = Number of bytes to be allocated
                                <1>      ;
                                <1>      ; OUTPUT ->
                                <1>      ;       1) cf = 0 -> successful
                                <1>      ;       EAX = Beginning (physical) address of the allocated memory block
                                <1>      ;       ECX = Number of allocated bytes (rounded up to page borders)
                                <1>      ;       2) cf = 1 -> unsuccessful
                                <1>      ;        2.1) If EAX > 0 ->
                                <1>      ;             (Number of requested pages is more than # of free pages
                                <1>      ;              but contiguous free pages -the aperture- is not enough!)
                                <1>      ;             EAX = Beginning address of available aperture
                                <1>      ;                  (one of all aperture with max. aperture size/length)
                                <1>      ;             ECX = Size of available aperture (memory block) in bytes
                                <1>      ;        2.2) If EAX = 0 -> Out of memory error
                                <1>      ;                   (number of free pages is less than requested number)
                                <1>      ;             ECX = Total number of free bytes (free pages * 4096)
                                <1>      ;                  (It is not number of contiguous free bytes)
                                <1>      ;
                                <1>      ; (Modified Registers -> EAX, ECX)
                                <1>      ;
                                <1>      ; PURPOSE: Loading a file at memory for copying or running etc.
                                <1>      ; If this procedure returns with cf is set, ECX contains maximum
                                <1>      ; available space and EAX contains the beginning address of it.
                                <1>      ; If EAX has zero, ECX contains total number of free bytes.
                                <1>      ; If requested block has been successfully allocated (by rounding up to
                                <1>      ; the last page border), it must be deallocated later by using
                                <1>      ; 'deallocate_memory_block' procedure.
                                <1>
00005D39 52                     <1>      push    edx ; *
00005D3A BAFF0F0000             <1>      mov     edx, PAGE_SIZE - 1   ; 4095
00005D3F 01D0                   <1>      add     eax, edx
00005D41 01D1                   <1>      add     ecx, edx
00005D43 C1E90C                 <1>      shr     ecx, PAGE_SHIFT            ; 12
                                <1>
                                <1>      ; ECX = number of contiguous pages to be allocated
00005D46 8B15[88760100]         <1>      mov     edx, [free_pages]
                                <1>      ; 01/05/2017
                                <1>      ;or     ecx, ecx
                                <1>      ;jz     short amb3
                                <1>      ; If ECX=0, set cf to 1 and return with max. available mem block size
                                <1>
00005D4C 39D1                   <1>      cmp     ecx, edx
00005D4E 7760                   <1>      ja      short amb_3
                                <1>
00005D50 C1E80C                 <1>      shr     eax, PAGE_SHIFT      ; 12
                                <1>
00005D53 89C2                   <1>      mov     edx, eax             ; page number
00005D55 C1EA03                 <1>      shr     edx, 3               ; to get offset to M.A.T.
                                <1>                                   ; (1 allocation bit = 1 page)
                                <1>                                   ; (1 allocation bytes = 8 pages)
00005D58 80E2FC                 <1>      and     dl, 0FCh             ; clear lower 2 bits
                                <1>                                   ; (to get 32 bit position)
00005D5B 53                     <1>      push    ebx ; **
                                <1> amb_0:
00005D5C 890D[3C820100]         <1>      mov     [mem_ipg_count], ecx ; initial (reset) value of page count
00005D62 890D[40820100]         <1>      mov     [mem_pg_count], ecx
00005D68 31C9                   <1>      xor     ecx, ecx ; 0
00005D6A 890D[44820100]         <1>      mov     [mem_aperture], ecx ; 0
00005D70 890D[48820100]         <1>      mov     [mem_max_aperture], ecx ; 0
                                <1>
00005D76 BB00001000             <1>      mov     ebx, MEM_ALLOC_TBL   ; Memory Allocation Table address.
00005D7B 3B15[8C760100]         <1>      cmp     edx, [next_page]     ; Is the beginning page address lower
                                <1>                                   ; than the address in 'next_page' ?
                                <1>                                   ; (the first/next free page of user space)
00005D81 7208                   <1>      jb      short amb_1
00005D83 3B15[90760100]         <1>      cmp     edx, [last_page]     ; is the beginning page address higher
                                <1>                                   ; than the address in 'last_page' ?
                                <1>                                   ; (end of the memory)
00005D89 7606                   <1>      jna     short amb_2          ; no
                                <1> amb_1:
00005D8B 8B15[8C760100]         <1>      mov     edx, [next_page]     ; M.A.T. offset (1 M.A.T. byte = 8 pages)
                                <1> amb_2:
00005D91 01D3                   <1>      add     ebx, edx
                                <1>
                                <1>      ; 28/04/2017
                                <1>      ;xor    ecx, ecx
00005D93 0FBC0B                 <1>      bsf     ecx, [ebx]           ; 0 to 31
00005D96 89D0                   <1>      mov     eax, edx
00005D98 C1E003                 <1>      shl     eax, 3               ; *8
00005D9B 01C8                   <1>      add     eax, ecx             ; beginning page number
                                <1>
00005D9D A3[4C820100]           <1>      mov     [mem_pg_pos], eax    ; beginning page no (for curr. mem. aperture)
00005DA2 A3[50820100]           <1>      mov     [mem_max_pg_pos], eax ; beginning page no for max. mem. aperture
                                <1>
00005DA7 83E01F                 <1>      and     eax, 1Fh             ; lower 5 bits only (0 to 31)
                                <1>                                   ; (allocation bit position)
00005DAA 750E                   <1>      jnz     short amb_4          ; 0
00005DAC B120                   <1>      mov     cl, 32
00005DAE EB4B                   <1>      jmp     short amb_10
                                <1>
                                <1> amb_3:       ; out_of_memory
00005DB0 31C0                   <1>      xor     eax, eax ; 0
00005DB2 89D1                   <1>      mov     ecx, edx ; free pages
00005DB4 C1E10C                 <1>      shl     ecx, PAGE_SHIFT
00005DB7 5A                     <1>      pop     edx ; *
00005DB8 F9                     <1>      stc
00005DB9 C3                     <1>      retn
                                <1> amb_4:
00005DBA 8B13                   <1>      mov     edx, [ebx]
00005DBC 88C1                   <1>      mov     cl, al ; 1 to 31
00005DBE D3EA                   <1>      shr     edx, cl
00005DC0 89D0                   <1>      mov     eax, edx
                                <1> amb_5:
00005DC2 D1E8                   <1>      shr     eax, 1 ; (***)
00005DC4 7317                   <1>      jnc     short amb_7
00005DC6 FF05[44820100]         <1>      inc     dword [mem_aperture]
00005DCC FF0D[40820100]         <1>      dec     dword [mem_pg_count]
00005DD2 747F                   <1>      jz      short amb_15
                                <1> amb_6:
                                <1>      ; 28/04/2017
00005DD4 FEC1                   <1>      inc     cl
00005DD6 80F920                 <1>      cmp     cl, 32
00005DD9 730D                   <1>      jnb     short amb_9
00005DDB EBE5                   <1>      jmp     short amb_5
                                <1> amb_7:
00005DDD 50                     <1>      push    eax ; (***) allocation bits (in shifted status)
00005DDE E828010000             <1>      call    amb_26 ; set maximum memory aperture (free memory block size)
00005DE3 58                     <1>      pop     eax ; (***)
00005DE4 EBEE                   <1>      jmp     short amb_6
                                <1> amb_8:
                                <1>      ; 28/04/2017
00005DE6 B120                   <1>      mov     cl, 32
                                <1> amb_9:
00005DE8 89DA                   <1>      mov     edx, ebx
```

```
2964 00005DEA 81EA00001000          <1>     sub     edx, MEM_ALLOC_TBL
2965 00005DF0 3B15[90760100]        <1>     cmp     edx, [last_page]
2966 00005DF6 7336                  <1>     jnb     short amb_14 ; contiguous pages not enough
2967 00005DF8 83C304                <1>     add     ebx, 4
2968                                <1> amb_10:
2969 00005DFB 8B03                  <1>     mov     eax, [ebx]
2970 00005DFD 21C0                  <1>     and     eax, eax
2971 00005DFF 7408                  <1>         jz      short amb_11 ; there is not a free page bit in this alloc dword
2972 00005E01 40                    <1>     inc     eax ; 0FFFFFFFFh -> 0
2973 00005E02 740C                  <1>     jz      short amb_12 ; all of bits are set (32 free pages)
2974 00005E04 48                    <1>     dec     eax
2975 00005E05 28C9                  <1>     sub     cl, cl ; 0
2976 00005E07 EBB9                  <1>     jmp     short amb_5
2977                                <1> amb_11:
2978 00005E09 E8FD000000            <1>     call    amb_26 ; set maximum memory aperture (free memory block size)
2979 00005E0E EBD8                  <1>     jmp     short amb_9
2980                                <1> amb_12:
2981 00005E10 390D[40820100]        <1>     cmp     [mem_pg_count], ecx ; 32
2982 00005E16 7306                  <1>     jnb     short amb_13
2983 00005E18 8B0D[40820100]        <1>     mov     ecx, [mem_pg_count]
2984                                <1> amb_13:
2985 00005E1E 010D[44820100]        <1>     add     [mem_aperture], ecx
2986 00005E24 290D[40820100]        <1>     sub     [mem_pg_count], ecx
2987 00005E2A 7627                  <1>     jna     short amb_15
2988 00005E2C EBBA                  <1>     jmp     short amb_9 ; 01/05/2017
2989                                <1> amb_14:
2990 00005E2E E8D8000000            <1>     call    amb_26 ; 28/04/2017
2991 00005E33 A1[50820100]          <1>     mov     eax, [mem_max_pg_pos] ; begin address of max. mem aperture
2992 00005E38 8B0D[48820100]        <1>     mov     ecx, [mem_max_aperture] ; max. (largest) memory aperture
2993 00005E3E F9                    <1>     stc
2994 00005E3F E9BC000000            <1>         jmp     amb_25
2995                                <1>
2996                                <1> allocate_lfb_memory_block:
2997                                <1>     ; 28/11/2023
2998                                <1>     ; 20/10/2023 - TRDOS 386 v2.0.7
2999                                <1>     ; (short way to set the LFB page bits on the M.A.T.)
3000                                <1>     ; called from 'allocate_lfb_pages_for_kernel'
3001 00005E44 A3[4C820100]          <1>     mov     [mem_pg_pos], eax    ; LFB start/base address as page number
3002 00005E49 890D[44820100]        <1>     mov     [mem_aperture], ecx  ; number of LFB pages
3003                                <1>                                  ; (!which overlapping main memory!)
3004 00005E4F 52                    <1>     push    edx ; *
3005 00005E50 53                    <1>     push    ebx ; **
3006 00005E51 EB0B                  <1>     jmp     short amb_16
3007                                <1>
3008                                <1> amb_15: ; OK !
3009 00005E53 A1[4C820100]          <1>     mov     eax, [mem_pg_pos]    ; Beginning address as page number
3010 00005E58 8B0D[44820100]        <1>     mov     ecx, [mem_aperture]  ; Free contiguous page count (>=1)
3011                                <1> amb_16:
3012                                <1>     ; allocate contiguous memory pages (via memory allocation table bits)
3013 00005E5E 89C2                  <1>     mov     edx, eax
3014                                <1>     ; 25/04/2017
3015 00005E60 C1EA03                <1>     shr     edx, 3          ; 8 pages in one allocation byte
3016 00005E63 80E2FC                <1>     and     dl, 0FCh        ; clear lower 2 bits
3017                                <1>                             ; (for dword/32bit positioning)
3018                                <1>
3019 00005E66 BB00001000            <1>     mov     ebx, MEM_ALLOC_TBL
3020 00005E6B 01D3                  <1>     add     ebx, edx
3021 00005E6D 83E01F                <1>     and     eax, 1Fh ; 31
3022                                <1>     ; 03/04/2016
3023 00005E70 BA20000000            <1>     mov     edx, 32
3024 00005E75 28C2                  <1>     sub     dl, al
3025 00005E77 39CA                  <1>     cmp     edx, ecx        ; ecx >= 1
3026 00005E79 7602                  <1>     jna     short amb_17
3027 00005E7B 89CA                  <1>     mov     edx, ecx
3028                                <1> amb_17:
3029 00005E7D 29D1                  <1>     sub     ecx, edx
3030 00005E7F 51                    <1>     push    ecx ; ***
3031 00005E80 89D1                  <1>     mov     ecx, edx
3032                                <1> amb_18:
3033 00005E82 0FB303                <1>     btr     [ebx], eax      ; The destination bit indexed by the source value
3034                                <1>                             ; is copied into the Carry Flag and then cleared
3035                                <1>                             ; in the destination.
3036                                <1>     ; 20/10/2023
3037                                <1>     ; (for LFB page allocation..
3038                                <1>     ;  -here, to prevent using LFB pages for another allocation later-
3039                                <1>     ;  /// the destination bit must be 0 -already- ..
3040                                <1>     ;  so, following 'jnc' is not necessary but i am not removing it
3041                                <1>     ;  for understanding why free page count is decreased here.)
3042                                <1>     ;
3043                                <1>     ;jnc    short amb_28 ; requested page is already allocated
3044                                <1>
3045 00005E85 FF0D[88760100]        <1>     dec     dword [free_pages] ; 1 page has been allocated (X = X-1)
3046                                <1> ;amb_28:    ; 20/10/2023
3047 00005E8B 49                    <1>     dec     ecx
3048 00005E8C 7404                  <1>     jz      short amb_19
3049 00005E8E FEC0                  <1>     inc     al
3050 00005E90 EBF0                  <1>     jmp     short amb_18
3051                                <1> amb_19:
3052 00005E92 59                    <1>     pop     ecx ; ***
3053                                <1>     ;and    ecx, ecx ; 0 ?
3054                                <1>     ;jz     short amb_22
3055                                <1>     ; 28/11/2023
3056 00005E93 E31E                  <1>     jecxz   amb_22
3057                                <1>     ; 01/04/2016
3058 00005E95 B020                  <1>     mov     al, 32
3059                                <1> amb_20:
3060 00005E97 83C304                <1>     add     ebx, 4
3061 00005E9A 39C1                  <1>     cmp     ecx, eax ; 32
3062 00005E9C 7305                  <1>     jnb     short amb_21
3063                                <1>     ; ECX < 32
3064 00005E9E 28C0                  <1>     sub     al, al ; 0
3065 00005EA0 50                    <1>     push    eax ; 0 ***
3066 00005EA1 EBDF                  <1>     jmp     short amb_18
3067                                <1> amb_21:
3068 00005EA3 2905[88760100]        <1>     sub     [free_pages], eax   ; [free_pages] = [free_pages] - 32
3069 00005EA9 C70300000000          <1>     mov     dword [ebx], 0     ; reset 32 bits
3070 00005EAF 29C1                  <1>     sub     ecx, eax ; 32
3071 00005EB1 75E4                  <1>     jnz     short amb_20
3072                                <1> amb_22:
3073 00005EB3 A1[4C820100]          <1>     mov     eax, [mem_pg_pos]   ; Beginning address as page number
3074 00005EB8 8B0D[44820100]        <1>     mov     ecx, [mem_aperture] ; Free contiguous page count
3075                                <1>     ; [next_page] update
3076 00005EBE 89C2                  <1>     mov     edx, eax
3077                                <1>     ; 03/04/2016
3078 00005EC0 C1EA03                <1>     shr     edx, 3              ; to get offset to M.A.T.
3079                                <1>                                 ; (1 allocation bit = 1 page)
3080                                <1>                                 ; (1 allocation bytes = 8 pages)
3081 00005EC3 80E2FC                <1>     and     dl, 0FCh            ; clear lower 2 bits
3082                                <1>                                 ; (to get 32 bit position)
3083 00005EC6 3B15[8C760100]        <1>     cmp     edx, [next_page] ; first free page pointer offset
3084 00005ECC 7732                  <1>     ja      short amb_25
3085 00005ECE BB00001000            <1>     mov     ebx, MEM_ALLOC_TBL
3086 00005ED3 833C1300              <1>     cmp     dword [ebx+edx], 0
3087 00005ED7 7721                  <1>     ja      short amb_24
```

```
3088 00005ED9 89C2                <1>     mov     edx, eax
3089 00005EDB 01CA                <1>     add     edx, ecx
3090 00005EDD C1EA03              <1>     shr     edx, 3
3091 00005EE0 80E2FC              <1>     and     dl, 0FCh
3092                              <1> amb_23:
3093 00005EE3 833C1300            <1>     cmp     dword [ebx+edx], 0
3094 00005EE7 7711                <1>     ja      short amb_24
3095 00005EE9 83C204              <1>     add     edx, 4
3096 00005EEC 3B15[90760100]      <1>     cmp     edx, [last_page]    ; last page pointer offset
3097 00005EF2 76EF                <1>     jna     short amb_23
3098 00005EF4 8B15[94760100]      <1>     mov     edx, [first_page]   ; (for) beginning of user's space
3099                              <1> amb_24:
3100 00005EFA 8915[8C760100]      <1>     mov     [next_page], edx
3101                              <1> amb_25:
3102 00005F00 9C                  <1>     pushf
3103 00005F01 C1E00C              <1>     shl     eax, PAGE_SHIFT             ; convert to phy. address in bytes
3104 00005F04 C1E10C              <1>     shl     ecx, PAGE_SHIFT             ; convert to byte counts
3105 00005F07 9D                  <1>     popf
3106 00005F08 5B                  <1>     pop     ebx ; **
3107 00005F09 5A                  <1>     pop     edx ; *
3108 00005F0A C3                  <1>     retn
3109                              <1>
3110                              <1> amb_26:     ; set maximum free memory aperture (free memory block size)
3111 00005F0B 89DA                <1>     mov     edx, ebx ; current address
3112 00005F0D 81EA00001000        <1>     sub     edx, MEM_ALLOC_TBL ; MAT beginning address
3113                              <1>     ; 02/04/2016
3114 00005F13 C1E203              <1>     shl     edx, 3 ; MAT byte offset * 8 = page number base
3115 00005F16 01CA                <1>     add     edx, ecx ; current page number (ecx = 0 to 32)
3116                              <1>     ;
3117 00005F18 A1[44820100]        <1>     mov     eax, [mem_aperture]
3118 00005F1D 21C0                <1>     and     eax, eax
3119 00005F1F 7421                <1>         jz      short amb_27
3120 00005F21 C705[44820100]0000- <1>         mov     dword [mem_aperture], 0
3120 00005F29 0000                <1>
3121 00005F2B 3B05[48820100]      <1>     cmp     eax, [mem_max_aperture]
3122 00005F31 760F                <1>     jna     short amb_27
3123 00005F33 A3[48820100]        <1>     mov     [mem_max_aperture], eax
3124                              <1>     ; 25/04/2017
3125 00005F38 A1[4C820100]        <1>     mov     eax, [mem_pg_pos]
3126                              <1>     ; EAX = Beginning page number of the max. aperture
3127 00005F3D A3[50820100]        <1>     mov     [mem_max_pg_pos], eax
3128                              <1> amb_27:
3129 00005F42 8915[4C820100]      <1>     mov     [mem_pg_pos], edx ; current page
3130                              <1>
3131 00005F48 A1[3C820100]        <1>     mov     eax, [mem_ipg_count] ; initial (reset) value of page count
3132 00005F4D A3[40820100]        <1>     mov     [mem_pg_count], eax
3133                              <1>
3134 00005F52 C3                  <1>     retn
3135                              <1>
3136                              <1> ; 04/06/2024
3137                              <1> %if 0
3138                              <1> deallocate_memory_block_x:
3139                              <1>     ; 26/11/2023 - TRDOS 386 v2.0.7
3140                              <1>     ; deallocate pages after rounding up
3141                              <1>     ; ((audio buffer size etc. may not be rounded up to page boundary))
3142                              <1>     add     ecx, PAGE_SIZE - 1   ; 4095
3143                              <1> %endif
3144                              <1>
3145                              <1> deallocate_memory_block:
3146                              <1>     ; 03/04/2016
3147                              <1>     ; 14/03/2016 (TRDOS 386 = TRDOS v2.0)
3148                              <1>     ; Deallocating contiguous memory pages (in the kernel's memory space)
3149                              <1>     ;
3150                              <1>     ; INPUT ->
3151                              <1>     ;       EAX = Beginning address (physical)
3152                              <1>     ;       ECX = Number of bytes to be deallocated
3153                              <1>     ;
3154                              <1>     ; OUTPUT ->
3155                              <1>     ;       Memory Allocation Table bits will be updated
3156                              <1>     ;       [free_pages] will be changed (increased)
3157                              <1>     ;
3158                              <1>     ; (Modified Registers -> EAX, ECX)
3159                              <1>     ;
3160                              <1>     ; PURPOSE: Unloading/Freeing a file -or an allocated memory block-
3161                              <1>     ; at memory after copying, running, saving, reading, writing etc.
3162                              <1>     ;
3163                              <1>
3164 00005F53 52                  <1>     push    edx ; *
3165 00005F54 53                  <1>     push    ebx ; **
3166                              <1>
3167 00005F55 C1E80C              <1>     shr     eax, PAGE_SHIFT             ; 12
3168 00005F58 C1E90C              <1>     shr     ecx, PAGE_SHIFT             ; 12
3169                              <1>
3170                              <1>     ; EAX = Beginning page number
3171                              <1>     ; ECX = Number of contiguous pages to be deallocated
3172                              <1> damb_0:
3173                              <1>     ; deallocate contiguous memory pages (via memory allocation table bits)
3174 00005F5B 89C2                <1>     mov     edx, eax
3175 00005F5D C1EA03              <1>     shr     edx, 3              ; to get offset to M.A.T.
3176                              <1>                                 ; (1 allocation bit = 1 page)
3177                              <1>                                 ; (1 allocation bytes = 8 pages)
3178 00005F60 80E2FC              <1>     and     dl, 0FCh            ; clear lower 2 bits
3179                              <1>                                 ; (to get 32 bit position)
3180 00005F63 3B15[8C760100]      <1>     cmp     edx, [next_page] ; next free page
3181 00005F69 7306                <1>     jnb     short damb_1
3182 00005F6B 8915[8C760100]      <1>     mov     [next_page], edx
3183                              <1> damb_1:
3184 00005F71 BB00001000          <1>     mov     ebx, MEM_ALLOC_TBL
3185 00005F76 01D3                <1>     add     ebx, edx
3186 00005F78 83E01F              <1>     and     eax, 1Fh ; 31
3187                              <1>
3188                              <1>     ; 03/04/2016
3189 00005F7B BA20000000          <1>     mov     edx, 32
3190 00005F80 28C2                <1>     sub     dl, al
3191 00005F82 39CA                <1>     cmp     edx, ecx
3192 00005F84 7602                <1>     jna     short damb_2
3193 00005F86 89CA                <1>     mov     edx, ecx
3194                              <1> damb_2:
3195 00005F88 29D1                <1>     sub     ecx, edx
3196 00005F8A 51                  <1>     push    ecx ; ***
3197 00005F8B 89D1                <1>     mov     ecx, edx
3198                              <1> damb_3:
3199 00005F8D 0FAB03              <1>     bts     [ebx], eax          ; unlink/release/deallocate page
3200                              <1>                                 ; set relevant bit to 1.
3201                              <1>                                 ; set CF to the previous bit value
3202 00005F90 FF05[88760100]      <1>     inc     dword [free_pages]   ; 1 page has been deallocated (X = X+1)
3203 00005F96 49                  <1>     dec     ecx
3204 00005F97 7404                <1>     jz      short damb_4
3205 00005F99 FEC0                <1>     inc     al
3206 00005F9B EBF0                <1>     jmp     short damb_3
3207                              <1> damb_4:
3208 00005F9D 59                  <1>     pop     ecx ; ***
3209 00005F9E 21C9                <1>     and     ecx, ecx ; 0 ?
3210 00005FA0 741E                <1>     jz      short damb_7
```

```
                                <1>     ; 03/04/2016
00005FA2 B020                   <1>     mov     al, 32
                                <1> damb_5:
00005FA4 83C304                 <1>     add     ebx, 4
00005FA7 39C1                   <1>     cmp     ecx, eax ; 32
00005FA9 7305                   <1>     jnb     short damb_6
                                <1>     ; ECX < 32
00005FAB 28C0                   <1>     sub     al, al ; 0
00005FAD 50                     <1>     push    eax ; 0 ***
00005FAE EBDD                   <1>     jmp     short damb_3
                                <1> damb_6:
00005FB0 0105[88760100]         <1>     add     [free_pages], eax ; [free_pages] = [free_pages] + 32
00005FB6 C703FFFFFFFF           <1>     mov     dword [ebx], 0FFFFFFFFh ; set 32 bits
00005FBC 29C1                   <1>     sub     ecx, eax ; 32
00005FBE 75E4                   <1>     jnz     short damb_5
                                <1> damb_7:
00005FC0 5B                     <1>     pop     ebx ; **
00005FC1 5A                     <1>     pop     edx ; *
00005FC2 C3                     <1>     retn
                                <1>
                                <1> direct_memory_access:
                                <1>     ; 20/10/2023
                                <1>     ; 17/04/2021 (temporary modifications)
                                <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
00005FC3 662500F0               <1>     and     ax, PTE_A_CLEAR ; clear page offset
00005FC7 50                     <1>     push    eax
                                <1>     ;and    ecx, ecx ; page count
                                <1>     ;jz     dmem_acc_7  ; 'insufficient memory' error
00005FC8 89C5                   <1>     mov     ebp, eax
00005FCA 81C300004000           <1>     add     ebx, CORE ; 12/05/2017
                                <1> dmem_acc_0:
00005FD0 891D[F08C0100]         <1>     mov     [base_addr], ebx ; 12/05/2017
00005FD6 A1[8C8E0100]           <1>     mov     eax, [u.pgdir] ; page dir address (physical)
00005FDB E858F9FFFF             <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 ?
00005FE0 7321                   <1>     jnc     short dmem_acc_1
                                <1>     ; 20/10/2023
                                <1>     ; Allocate a page as a new page table
00005FE2 E83FF8FFFF             <1>     call    allocate_page
                                <1>     ;jc     dmem_acc_7  ; 'insufficient memory' error
                                <1>     ; 17/04/2021
00005FE7 7215                   <1>     jc      short _dmem_acc_7
                                <1>     ;
00005FE9 E8A9F8FFFF             <1>     call    clear_page
                                <1>     ; EAX = Physical (base) address of the allocated (new) page
00005FEE 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)
00005FF0 8902                   <1>     mov     [edx], eax ; Let's put the new page directory entry here !
00005FF2 A1[8C8E0100]           <1>     mov     eax, [u.pgdir]
00005FF7 E83CF9FFFF             <1>     call    get_pte
                                <1>         ;jc dmem_acc_7 ; 'insufficient memory' error
                                <1>     ; 17/04/2021
00005FFC 7305                   <1>     jnc     short dmem_acc_1
                                <1> _dmem_acc_7:
00005FFE E985000000             <1>     jmp     dmem_acc_7
                                <1> dmem_acc_1:
                                <1>     ; EAX = PTE value, EDX = PTE address
00006003 A801                   <1>     test    al, PTE_A_PRESENT
                                <1>     ;jnz    short dmem_acc_2   ; 17/04/2021 (*)
                                <1>     ; 17/04/2021 (temporary)
00006005 745F                   <1>     jz      short short dmem_acc_6  ; ! temporary ! (*)
                                <1>     ; 20/10/2023
                                <1>     ; If the page table entry is zero or invalid -not present-
                                <1>     ; it will be directy allocated for the physical memory section
                                <1>     ; without using the M.A.T. (Specially for LFB access -generally-
                                <1>     ; is at the beyod of the physical main memory end.)
                                <1>     ; (M.A.T bits are not used for direct memory access. To cancel
                                <1>     ; direct memory access also is out of the M.A.T scope because
                                <1>     ; direct accessed physical pages are marked as SHARED on their
                                <1>     ; PTE. Shared pages are not dealloaced.)
                                <1>
                                <1> ; 17/04/2021
                                <1> ; (following code is disabled as temporary)
                                <1> ;
```

```
                            <1> ;   or      eax, eax
                            <1> ;   jz      short dmem_acc_6   ; Change PTE
                            <1> ;   shr     eax, 1          ; swap disk block (8 sectors) address
                            <1> ;   ; unlink swap disk block
                            <1> ;   call    unlink_swap_block
                            <1> ;   jmp     short dmem_acc_6
                            <1>
                            <1> dmem_acc_2:
00006007 A802               <1>     test    al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
                            <1>                               ; (must be 1)
00006009 7550               <1>     jnz     short dmem_acc_4
                            <1>     ; Read only -duplicated- page (belongs to a parent or a child)
0000600B 66A90002           <1>         test    ax, PTE_DUPLICATED ; Was this page duplicated
                            <1>                                    ; as child's page ?
0000600F 7455               <1>     jz      short dmem_acc_5 ; Change PTE but don't deallocate the page!
                            <1>
                            <1>     ;push   edi
                            <1>     ;push   esi
                            <1>
00006011 51                 <1>     push    ecx
                            <1>     ;push   ebx
00006012 8B1D[908E0100]     <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..
00006018 89EF               <1>     mov     edi, ebp
0000601A C1EF16             <1>     shr     edi, PAGE_D_SHIFT ; 22
                            <1>     ; EDI = page directory entry index (0-1023)
0000601D 89EE               <1>     mov     esi, ebp
0000601F C1EE0C             <1>     shr     esi, PAGE_SHIFT ; 12
00006022 81E6FF030000       <1>     and     esi, PTE_MASK
                            <1>     ; ESI = page table entry index (0-1023)
                            <1>
00006028 66C1E702           <1>     shl     di, 2 ; * 4
0000602C 01FB               <1>     add     ebx, edi ; PDE offset (for the parent)
0000602E 8B0F               <1>     mov     ecx, [edi]
00006030 F6C101             <1>     test    cl, PDE_A_PRESENT ; present (valid) or not ?
00006033 7425               <1>     jz      short dmem_acc_3      ; parent process does not use this page
00006035 6681E100F0         <1>     and     cx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
0000603A 66C1E602           <1>     shl     si, 2 ; *4
0000603E 01CE               <1>     add     esi, ecx ; PTE offset (for the parent)
00006040 8B1E               <1>     mov     ebx, [esi]
00006042 F6C301             <1>     test    bl, PTE_A_PRESENT ; present or not ?
00006045 7413               <1>     jz      short dmem_acc_3      ; parent process does not use this page
00006047 662500F0           <1>     and     ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
0000604B 6681E300F0         <1>     and     bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
00006050 39D8               <1>     cmp     eax, ebx       ; parent's and child's pages are same ?
00006052 7506               <1>     jne     short dmem_acc_3       ; not same page
                            <1>                                    ; deallocate the child's page
00006054 800E02             <1>         or      byte [esi], PTE_A_WRITE ; convert to writable page (parent)
                            <1>     ;pop    ebx
00006057 59                 <1>     pop     ecx
00006058 EB0C               <1>     jmp     short dmem_acc_5
                            <1> dmem_acc_3:
                            <1>     ;pop    ebx
0000605A 59                 <1>     pop     ecx
                            <1> dmem_acc_4:
0000605B 66A90004           <1>     test    ax, PTE_SHARED ; shared or direct memory access indicator
0000605F 7505               <1>     jnz     short dmem_acc_5   ; AVL bit 1 = 1, do not deallocate this page!
                            <1>     ;
                            <1>     ;and    ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
00006061 E87FF9FFFF         <1>     call    deallocate_page
                            <1> dmem_acc_5:
                            <1>     ;pop    esi
                            <1>     ;pop    edi
                            <1> dmem_acc_6:
00006066 89E8               <1>     mov     eax, ebp ; physical page (offset=0) address
                            <1>     ; EAX = memory page address
                            <1>     ; EDX = PTE entry address (physical)
00006068 660D0704           <1>     or      ax, PTE_A_PRESENT+PTE_A_USER+PTE_A_WRITE+PTE_SHARED
                            <1>                     ; present flag, bit 0 = 1
                            <1>                     ; user flag, bit 2 = 1
                            <1>                     ; writable flag, bit 1 = 1
                            <1>                     ; direct memory access flag, bit 10 = 1
                            <1>                     ; (This page must not be deallocated!)
0000606C 8902               <1>     mov     [edx], eax  ; Update PTE value
0000606E 49                 <1>     dec     ecx ; remain count of contiguous pages
0000606F 741E               <1>     jz      short dmem_acc_8
00006071 81C500100000       <1>     add     ebp, PAGE_SIZE ; next physical page address
                            <1>     ; 22/07/2017
                            <1>     ;mov    eax, ebp
                            <1>     ; 12/05/2017
00006077 8B1D[F08C0100]     <1>     mov     ebx, [base_addr] ; linear address (virtual+CORE)
0000607D 81C300100000       <1>     add     ebx, PAGE_SIZE ; next linear address
00006083 E948FFFFFF         <1>         jmp     dmem_acc_0
                            <1> dmem_acc_7:  ; ERROR !
00006088 C7042404000000     <1>     mov     dword [esp], ERR_MINOR_IM
                            <1>             ; Insufficient memory (minor) error!
                            <1>             ; Major error = 0 (No protection fault)
                            <1>     ; cf = 1
                            <1> dmem_acc_8:
0000608F 58                 <1>     pop     eax
                            <1>     ;pop    edx
                            <1>     ;pop    ecx
                            <1>     ;pop    ebx
                            <1>     ;pop    ebp
00006090 C3                 <1>     retn
                            <1>
                            <1> deallocate_user_pages:
                            <1>     ; 20/05/2017
                            <1>     ; 15/05/2017
                            <1>     ; 20/02/2017
                            <1>     ; 19/02/2017 (TRDOS 386 = TRDOS v2.0)
                            <1>     ;
                            <1>     ; Deallocate virtually contiguous user pages (memory block)
                            <1>     ; (caller: 'sysdalloc' system call)
                            <1>     ;
                            <1>     ; INPUT ->
                            <1>     ;       EBX = VIRTUAL ADDRESS (beginning address)
                            <1>     ;       ECX = byte count
                            <1>     ;       [u.pgdir] = user's page directory
                            <1>     ;       [u.ppdir] = parent's page directory
                            <1>     ;
                            <1>     ; OUTPUT ->
                            <1>     ;    If CF = 0
                            <1>     ;       EAX = Deallocated memory bytes
                            <1>     ;         (Even if shared or read only pages will not be
                            <1>     ;          deallocated on M.A.T., this byte count will be
                            <1>     ;          returned as virtually deallocated bytes; in fact
                            <1>     ;          virtually deallocated user pages * 4096.)
                            <1>     ;       EBX = Virtual address (as rounded up)
                            <1>     ;    If CF = 1
                            <1>     ;       EAX = 0 (there is not any deallocated pages)
                            <1>     ;
                            <1>     ; Note: Empty page tables will not be deallocated!!!
```

```
                        <1>     ;     (they will be deallocated at process termination stage)
                        <1>     ;
                        <1>     ; Modified Registers -> EAX, EDX, ESI, EDI, EBX, ECX, EBP
                        <1>     ;
00006091 89DE           <1>     mov     esi, ebx
00006093 89F7           <1>     mov     edi, esi
00006095 01CF           <1>     add     edi, ecx
00006097 81C6FF0F0000   <1>     add     esi, PAGE_SIZE - 1  ; 4095 (round up)
0000609D C1EE0C         <1>     shr     esi, PAGE_SHIFT
000060A0 C1EF0C         <1>     shr     edi, PAGE_SHIFT
000060A3 89F8           <1>     mov     eax, edi ; end page
000060A5 29F0           <1>     sub     eax, esi ; end page - start page
000060A7 0F86C8000000   <1>     jna     da_u_pd_err  ; < 1
000060AD 89F3           <1>     mov     ebx, esi
000060AF C1E30C         <1>     shl     ebx, PAGE_SHIFT ; virtual address (as rounded up)
000060B2 53             <1>     push    ebx ; *
000060B3 89C1           <1>     mov     ecx, eax ; page count
000060B5 C1E00C         <1>     shl     eax, PAGE_SHIFT ; byte count as adjusted
000060B8 50             <1>     push    eax ; **
000060B9 8B1D[8C8E0100] <1>     mov     ebx, [u.pgdir] ; physical addr of user's page dir
000060BF 81C600040000   <1>     add     esi, CORE/PAGE_SIZE
000060C5 89F7           <1>     mov     edi, esi
000060C7 81E7FF030000   <1>     and     edi, PTE_MASK ; PTE entry in the page table
000060CD 57             <1>     push    edi ; *** ; PTE index (of page directory)
000060CE C1EE0A         <1>     shr     esi, PAGE_D_SHIFT - PAGE_SHIFT ; 22-12=10
000060D1 89F2           <1>     mov     edx, esi
                        <1>     ; EDX = PDE index
000060D3 C1E602         <1>     shl     esi, 2 ; convert PDE index to dword offset
000060D6 01DE           <1>     add     esi, ebx ; add page directory address
                        <1> da_u_pd_1:
000060D8 AD             <1>     lodsd
                        <1>     ;
000060D9 89F5           <1>     mov     ebp, esi ; 20/02/2017
                        <1>     ; EBP = next PDE address
                        <1>     ;
000060DB A801           <1>     test    al, PDE_A_PRESENT ; bit 0, present flag (must be 1)
000060DD 0F8487000000   <1>     jz      da_u_pd_3 ; 20/05/2017
000060E3 662500F0       <1>     and     ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
                        <1>     ; EAX = PHYSICAL (flat) ADDRESS OF THE PAGE TABLE
000060E7 8B3C24         <1>     mov     edi, [esp] ; ***
                        <1>     ; EDI = PTE index (of complete page directory)
                        <1>     ;and    edi, PTE_MASK ; PTE entry in the page table
000060EA C1E702         <1>     shl     edi, 2 ; convert PTE index to dword offset
000060ED 89FE           <1>     mov     esi, edi ; PTE offset in page table (0-4092)
000060EF 01C6           <1>     add     esi, eax ; now, esi points to requested PTE
                        <1> da_u_pt_0:
000060F1 AD             <1>     lodsd
000060F2 A801           <1>     test    al, PTE_A_PRESENT ; bit 0, present flag (must be 1)
000060F4 7452           <1>     jz      short da_u_pt_1
                        <1>     ;
000060F6 A802           <1>     test    al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
                        <1>                               ; (must be 1)
000060F8 753C           <1>     jnz     short da_u_pt_3
                        <1>     ; Read only -duplicated- page (belongs to a parent or a child)
000060FA 66A90002       <1>         test    ax, PTE_DUPLICATED ; Was this page duplicated
                        <1>                                    ; as child's page ?
000060FE 7441           <1>     jz      short da_u_pt_4 ; Clear PTE but don't deallocate the page!
                        <1>     ;
                        <1>     ; check the parent's PTE value is read only & same page or not..
                        <1>     ; EDX = page directory entry index (0-1023)
00006100 52             <1>     push    edx ; ****
                        <1>     ; EDI = page table entry offset (0-4092)
00006101 8B1D[908E0100] <1>     mov     ebx, [u.ppgdir] ; page directory of the parent process
00006107 66C1E202       <1>     shl     dx, 2 ; *4
0000610B 01D3           <1>     add     ebx, edx ; PDE address (for the parent)
0000610D 8B13           <1>     mov     edx, [ebx] ; page table address
0000610F F6C201         <1>     test    dl, PDE_A_PRESENT ; present (valid) or not ?
00006112 7421           <1>     jz      short da_u_pt_2       ; parent process does not use this page
00006114 6681E200F0     <1>     and     dx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
                        <1>     ; EDI = page table entry offset (0-4092)
00006119 01D7           <1>     add     edi, edx       ; PTE address (for the parent)
0000611B 8B1F           <1>     mov     ebx, [edi]
0000611D F6C301         <1>     test    bl, PTE_A_PRESENT ; present or not ?
00006120 7413           <1>     jz      short da_u_pt_2       ; parent process does not use this page
00006122 662500F0       <1>     and     ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
00006126 6681E300F0     <1>     and     bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
0000612B 39D8           <1>     cmp     eax, ebx       ; parent's and child's pages are same ?
0000612D 7506           <1>     jne     short da_u_pt_2       ; not same page
                        <1>                               ; deallocate the child's page
0000612F 800F02         <1>         or      byte [edi], PTE_A_WRITE ; convert to writable page (parent)
00006132 5A             <1>     pop     edx ; ****
00006133 EB0C           <1>     jmp     short da_u_pt_4
                        <1>
                        <1> ; 17/04/2021
                        <1> ; ('da_u_pt_1' is disabled as temporary)
                        <1>
                        <1> ;da_u_pt_1:
                        <1> ;   or      eax, eax       ; swapped page ?
                        <1> ;   jz      short da_u_pt_5       ; no
                        <1> ;                         ; yes
                        <1> ;   shr     eax, 1
                        <1> ;   call    unlink_swap_block ; Deallocate swapped page block
                        <1> ;                             ; on the swap disk (or in file)
                        <1> ;   jmp     short da_u_pt_5
                        <1> da_u_pt_2:
00006135 5A             <1>     pop     edx ; ****
                        <1> da_u_pt_3:
00006136 66A90004       <1>     test    ax, PTE_SHARED ; shared or direct memory access indicator
0000613A 7505           <1>     jnz     short da_u_pt_4       ; AVL bit 1 = 1, do not deallocate this page!
                        <1>     ;
                        <1>     ;and    ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
0000613C E8A4F8FFFF     <1>     call    deallocate_page ; set the mem allocation bit of this page
                        <1> da_u_pt_4:
00006141 C746FC00000000 <1>     mov     dword [esi-4], 0 ; clear/reset PTE (child, dupl. as parent)
                        <1> ; 17/04/2021 (temporary)
                        <1> da_u_pt_1:
                        <1> da_u_pt_5:
                        <1>     ; 20/05/2017
00006148 58             <1>     pop     eax ; *** PTE index (of page directory)
00006149 49             <1>     dec     ecx ; remain page count
0000614A 7426           <1>     jz      short da_u_pd_4
0000614C 40             <1>     inc     eax ; next PTE
0000614D 6625FF03       <1>     and     ax, PTE_MASK ; PTE entry index in the page table
00006151 50             <1>     push    eax ; *** (save again)
                        <1>     ;mov    edi, eax
                        <1>     ;and    di, PTE_MASK
                        <1>     ;cmp    edi, PAGE_SIZE / 4 ; 1024
                        <1>     ;jnb    short da_u_pd_2
00006152 89C7           <1>     mov     edi, eax
00006154 C1E702         <1>     shl     edi, 2 ; convert index to dword offset
                        <1>     ;test   ax, PTE_MASK ; 3FFh
00006157 09C0           <1>     or      eax, eax
00006159 7596           <1>     jnz     short da_u_pt_0 ; 1-1023
                        <1> da_u_pd_2:
```

```
3583 0000615B 42                <1>     inc     edx
3584                            <1>     ; 20/05/2017
3585 0000615C 6681E2FF03        <1>     and     dx, PTE_MASK  ; 3FFh
3586 00006161 740F              <1>     jz      short da_u_pd_4  ; 0 (1024)
3587                            <1>     ;cmp    edx, 1024
3588                            <1>     ;jnb    short da_u_pd_4
3589 00006163 89EE              <1>     mov     esi, ebp ; 20/02/2017
3590 00006165 E96EFFFFFF        <1>     jmp     da_u_pd_1
3591                            <1> da_u_pd_3:
3592                            <1>     ; 15/05/2017 (empty page directory entry)
3593 0000616A 81E900040000      <1>     sub     ecx, 1024
3594 00006170 77E9              <1>     ja      short da_u_pd_2 ; 20/05/2017
3595                            <1> da_u_pd_4:
3596 00006172 58                <1>     pop     eax ; **
3597 00006173 5B                <1>     pop     ebx ; *
3598 00006174 C3                <1>     retn
3599                            <1>
3600                            <1> da_u_pd_err:
3601 00006175 31C0              <1>     xor     eax, eax
3602 00006177 F9                <1>     stc
3603 00006178 C3                <1>     retn
3604                            <1>
3605                            <1> allocate_user_pages:
3606                            <1>     ; 20/05/2017
3607                            <1>     ; 01/05/2017, 02/05/2017, 15/05/2017
3608                            <1>     ; 04/03/2017
3609                            <1>     ; 20/02/2017 (TRDOS 386 = TRDOS v2.0)
3610                            <1>     ;
3611                            <1>     ; Allocate physically contiguous user pages (memory block)
3612                            <1>     ; (caller: 'sysalloc' system call)
3613                            <1>     ;
3614                            <1>     ; Note: This procedure does not alloc a page's itself
3615                            <1>     ;       (page bits) on Memory Allocation Table.
3616                            <1>     ;       (allocate_memory_block is needed before this proc)
3617                            <1>     ;
3618                            <1>     ; INPUT ->
3619                            <1>     ;       EAX = PHYSICAL ADDRESS (beginning address)
3620                            <1>     ;       EBX = VIRTUAL ADDRESS (beginning address)
3621                            <1>     ;       ECX = byte count (>=4096)
3622                            <1>     ;       [u.pgdir] = user's page directory
3623                            <1>     ;
3624                            <1>     ;       Note: All addresses are (must be) already adjusted
3625                            <1>     ;       to page borders, otherwise, lower 12bits of addresses
3626                            <1>     ;       and byte count would be truncated.
3627                            <1>     ;
3628                            <1>     ; OUTPUT ->
3629                            <1>     ;       none
3630                            <1>     ;
3631                            <1>     ;       CF = 1 -> insufficient memory error
3632                            <1>     ;
3633                            <1>     ; Note: All pages will be allocated in physical page order
3634                            <1>     ;       from the beginning page address.
3635                            <1>     ;       * A new page table will be added to the page dir
3636                            <1>     ;         when the requested PDE is invalid.
3637                            <1>     ;       * Those pages will not be added to swap queue
3638                            <1>     ;         because main purpose of this allocation is to
3639                            <1>     ;         set a direct memory access (DMA controller) buffer.
3640                            <1>     ;        (Swapping out a page in a DMA buffer would be wrong!)
3641                            <1>     ;       * Previous content of page tables (PTEs) would be
3642                            <1>     ;         (should be) deallocated before entering this
3643                            <1>     ;         procedure. So, new page table entries (PTEs)
3644                            <1>     ;         directly will be written without checking
3645                            <1>     ;         their previous content.
3646                            <1>     ;       * Only solution to increase free memory by removing
3647                            <1>     ;         that non-swappable memory block is to terminate
3648                            <1>     ;         the process or to wait until the process will
3649                            <1>     ;         deallocate that memory block as itself. ('sysdalloc')
3650                            <1>     ;         (No problem, if the process does not grab all of
3651                            <1>     ;         -very big amount of- free memory by using
3652                            <1>     ;         'sysalloc' system call!?)
3653                            <1>     ;         (Even if the process has grabbed all of free memory,
3654                            <1>     ;         no problem if the process is not running in
3655                            <1>     ;         multitasking mode. No problem in multitasking
3656                            <1>     ;         mode if there is not another process which is running
3657                            <1>     ;         or waiting or sleeping for an event as it's pages
3658                            <1>     ;         are swapped-out. But a new process can not start to
3659                            <1>     ;         run if all of free memory has beeen allocated
3660                            <1>     ;         by running processes. Deallocation -'sysdalloc'-
3661                            <1>     ;         or terminate a running process is needed
3662                            <1>     ;         in order to run a new process.)
3663                            <1>     ;
3664                            <1>     ; Modified Registers -> EAX, EDX, ESI, EDI, EBX, ECX, EBP
3665                            <1>     ;
3666                            <1>
3667                            <1>     ; 01/05/2017
3668 00006179 662500F0          <1>     and     ax, ~PAGE_OFF
3669 0000617D 6681E300F0        <1>     and     bx, ~PAGE_OFF
3670                            <1>     ; 02/05/2017
3671 00006182 BD00F0FFFF        <1>     mov     ebp, 0FFFFF000h ; 4 Giga Bytes - 4096 Bytes (for Stack)
3672 00006187 C1E90C            <1>     shr     ecx, PAGE_SHIFT ; page count
3673 0000618A 83F901            <1>     cmp     ecx, 1
3674 0000618D 7251              <1>     jb      short a_u_im_retn
3675 0000618F 89C2              <1>     mov     edx, eax
3676 00006191 01CA              <1>     add     edx, ecx
3677 00006193 724B              <1>     jc      short a_u_im_retn
3678 00006195 39D5              <1>     cmp     ebp, edx
3679 00006197 7247              <1>     jb      short a_u_im_retn
3680 00006199 89DA              <1>     mov     edx, ebx
3681 0000619B 81C200004000      <1>     add     edx, CORE
3682 000061A1 723D              <1>     jc      short a_u_im_retn
3683 000061A3 01CA              <1>     add     edx, ecx
3684 000061A5 7239              <1>     jc      short a_u_im_retn
3685 000061A7 39D5              <1>     cmp     ebp, edx
3686 000061A9 7235              <1>     jb      short a_u_im_retn
3687                            <1>     ;
3688 000061AB 89C5              <1>     mov     ebp, eax ; physical address
3689 000061AD 89DE              <1>     mov     esi, ebx
3690 000061AF 81C600004000      <1>     add     esi, CORE ; start of user's memory (4M)
3691 000061B5 C1EE0C            <1>     shr     esi, PAGE_SHIFT ; higher 20 bits of the linear address
3692                            <1>     ;shr    ecx, PAGE_SHIFT ; page count
3693 000061B8 8B1D[8C8E0100]    <1>     mov     ebx, [u.pgdir] ; physical addr of user's page dir
3694 000061BE 89F7              <1>     mov     edi, esi
3695 000061C0 81E7FF030000      <1>     and     edi, PTE_MASK ; PTE entry index in the page table
3696 000061C6 57                <1>     push    edi  ; * ; PTE index (in page directory)
3697 000061C7 C1EE0A            <1>     shr     esi, PAGE_D_SHIFT - PAGE_SHIFT ; 22-12=10
3698 000061CA 89F2              <1>     mov     edx, esi
3699                            <1>     ; EDX = PDE index
3700 000061CC C1E602            <1>     shl     esi, 2 ; convert PDE index to dword offset
3701 000061CF 01DE              <1>     add     esi, ebx ; add page directory address
3702                            <1> a_u_pd_0:
3703 000061D1 AD                <1>     lodsd
3704                            <1>     ;
3705 000061D2 89F3              <1>     mov     ebx, esi ; next PDE address
3706                            <1>     ;
```

```
3707 000061D4 A801              <1>     test    al, PDE_A_PRESENT ; bit 0, present flag (must be 1)
3708 000061D6 7513              <1>     jnz     short a_u_pd_2
3709                            <1>     ;
3710                            <1>     ; empty PDE (it does not point to valid page table address)
3711 000061D8 E849F6FFFF        <1>     call    allocate_page  ; (allocate a new page table)
3712 000061DD 7302              <1>     jnc     short a_u_pd_1 ; OK... now, we have a new page table.
3713                            <1>     ; cf = 1
3714                            <1>     ; There is not a free memory page to allocate a new page table !!!
3715 000061DF 5E                <1>     pop     esi ; *
3716                            <1> a_u_im_retn:
3717 000061E0 C3                <1>     retn    ; return to 'sysalloc' with 'insufficient memory' error
3718                            <1>     ;
3719                            <1> a_u_pd_1: ; clear the new page table content
3720                            <1>     ; EAX = Physical (base) address of the new page table
3721 000061E1 E8B1F6FFFF        <1>     call    clear_page ; Clear page content
3722                            <1>     ;
3723 000061E6 0C07              <1>     or      al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER
3724                            <1>              ; set bit 0, bit 1 and bit 2 to 1
3725                            <1>              ; (present, writable, user)
3726 000061E8 8946FC            <1>     mov     [esi-4], eax
3727                            <1> a_u_pd_2:
3728 000061EB 662500F0          <1>     and     ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
3729                            <1>     ; EAX = PHYSICAL (flat) ADDRESS OF THE PAGE TABLE
3730 000061EF 8B3C24            <1>     mov     edi, [esp] ; *
3731                            <1>     ; EDI = PTE index (of page directory)
3732                            <1>     ;and    edi, PTE_MASK ; PTE entry index in the page table
3733                            <1>     ; EBX = next PDE address
3734 000061F2 89FE              <1>     mov     esi, edi ; PTE index in page table (0-1023)
3735 000061F4 C1E702            <1>     shl     edi, 2 ; convert PTE index to dword offset
3736 000061F7 01C7              <1>     add     edi, eax ; now, edi points to requested PTE
3737                            <1> a_u_pt_0:
3738                            <1>     ; 02/05/2017
3739 000061F9 8B07              <1>     mov     eax, [edi]
3740                            <1>     ;
3741 000061FB A801              <1>     test    al, PTE_A_PRESENT ; bit 0, present flag (must be 1)
3742 000061FD 7445              <1>     jz      short a_u_pt_1
3743                            <1>     ;
3744 000061FF A802              <1>     test    al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
3745                            <1>                               ; (must be 1)
3746 00006201 7550              <1>     jnz     short a_u_pt_3
3747                            <1>     ; Read only -duplicated- page (belongs to a parent or a child)
3748 00006203 66A90002          <1>         test    ax, PTE_DUPLICATED ; Was this page duplicated
3749                            <1>                                    ; as child's page ?
3750 00006207 7455              <1>     jz      short a_u_pt_4 ; Clear PTE but don't deallocate the page!
3751                            <1>     ;
3752                            <1>     ; check the parent's PTE value is read only & same page or not..
3753                            <1>     ; EDX = page directory entry index (0-1023)
3754 00006209 52                <1>     push    edx ; **
3755 0000620A 53                <1>     push    ebx ; ***
3756                            <1>     ; ESI = page table entry index (0-1023)
3757                            <1>     ;push   esi ; **** ; 20/05/2017
3758 0000620B 8B1D[908E0100]    <1>     mov     ebx, [u.ppgdir] ; page directory of the parent process
3759 00006211 66C1E202          <1>     shl     dx, 2 ; *4
3760 00006215 01D3              <1>     add     ebx, edx ; PTE address,0 (for the parent)
3761 00006217 8B13              <1>     mov     edx, [ebx] ; page table address
3762 00006219 F6C201            <1>     test    dl, PDE_A_PRESENT ; present (valid) or not ?
3763 0000621C 7433              <1>     jz      short a_u_pt_2 ; parent process does not use this page
3764 0000621E 6681E200F0        <1>     and     dx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
3765 00006223 66C1E602          <1>     shl     si, 2 ; *4
3766                            <1>     ; ESI = page table entry offset (0-4092)
3767 00006227 01D6              <1>     add     esi, edx        ; PTE address (for the parent)
3768 00006229 8B1E              <1>     mov     ebx, [esi]
3769 0000622B F6C301            <1>     test    bl, PTE_A_PRESENT ; present or not ?
3770 0000622E 7421              <1>     jz      short a_u_pt_2 ; parent process does not use this page
3771 00006230 662500F0          <1>     and     ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
3772 00006234 6681E300F0        <1>     and     bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
3773 00006239 39D8              <1>     cmp     eax, ebx        ; parent's and child's pages are same ?
3774 0000623B 7514              <1>     jne     short a_u_pt_2 ; not same page
3775                            <1>                             ; deallocate the child's page
3776 0000623D 800E02            <1>         or      byte [esi], PTE_A_WRITE ; convert to writable page (parent)
3777                            <1>     ;pop    esi ; **** ; 20/05/2017
3778 00006240 5B                <1>     pop     ebx ; ***
3779 00006241 5A                <1>     pop     edx ; **
3780 00006242 EB1A              <1>     jmp     short a_u_pt_4
3781                            <1> a_u_pt_1:
3782 00006244 09C0              <1>     or      eax, eax        ; swapped page ?
3783 00006246 7416              <1>     jz      short a_u_pt_4 ; no
3784                            <1>                             ; yes
3785 00006248 D1E8              <1>     shr     eax, 1
3786 0000624A E8B5F9FFFF        <1>     call    unlink_swap_block ; Deallocate swapped page block
3787                            <1>                               ; on the swap disk (or in file)
3788 0000624F EB0D              <1>     jmp     short a_u_pt_4
3789                            <1> a_u_pt_2:
3790                            <1>     ;pop    esi ; **** ; 20/05/2017
3791 00006251 5B                <1>     pop     ebx ; ***
3792 00006252 5A                <1>     pop     edx ; **
3793                            <1> a_u_pt_3:
3794 00006253 66A90004          <1>     test    ax, PTE_SHARED ; shared or direct memory access indicator
3795 00006257 7505              <1>     jnz     short a_u_pt_4 ; AVL bit 1 = 1, do not deallocate this page!
3796                            <1>     ;
3797                            <1>     ;and    ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
3798 00006259 E887F7FFFF        <1>     call    deallocate_page ; set the mem allocation bit of this page
3799                            <1>     ;
3800                            <1> a_u_pt_4:
3801 0000625E 89E8              <1>     mov     eax, ebp ; physical address
3802 00006260 0C07              <1>     or      al, PTE_A_PRESENT + PTE_A_WRITE + PTE_A_USER ; 04/03/2017
3803 00006262 AB                <1>     stosd
3804 00006263 5E                <1>     pop     esi ; * ; 20/05/2017
3805 00006264 49                <1>     dec     ecx ; remain page count
3806 00006265 7417              <1>     jz      short a_u_pd_5
3807 00006267 81C500100000      <1>     add     ebp, PAGE_SIZE
3808 0000626D 46                <1>     inc     esi ; next PTE (index)
3809                            <1>     ; 20/05/2017
3810                            <1>     ;cmp    esi, PAGE_SIZE/4 ; 1024
3811                            <1>     ;jb     short a_u_pt_0
3812 0000626E 6681E6FF03        <1>     and     si, PTE_MASK ; 3FFh (0 to 1023)
3813 00006273 56                <1>     push    esi ; *
3814 00006274 7583              <1>     jnz     short a_u_pt_0 ; > 0 (<1024)
3815                            <1> a_u_pd_3:
3816 00006276 42                <1>     inc     edx
3817                            <1> ;   cmp     edx, 1024
3818                            <1> ;   jnb     short a_u_pd_4 ; 02/05/2017 (error!, ecx > 0)
3819 00006277 89DE              <1>     mov     esi, ebx ; the next PDE address
3820 00006279 E953FFFFFF        <1>     jmp     a_u_pd_0
3821                            <1> a_u_pd_4:
3822                            <1>     ; 02/05/2017
3823                            <1> ;   stc
3824                            <1> a_u_pd_5:
3825                            <1>     ; 20/05/2017
3826                            <1>     ;pop    edi ; *
3827 0000627E C3                <1>     retn
3828                            <1>
3829                            <1>     ; 28/11/2023
3830                            <1> ;a_lfb_k__err:
```

```
3831                                <1>     ;retn
3832                                <1>
3833                                <1> allocate_lfb_pages_for_kernel:
3834                                <1>     ; 02/12/2023
3835                                <1>     ; 29/11/2023
3836                                <1>     ; 28/11/2023
3837                                <1>     ; 20/10/2023 - TRDOS 386 v2.0.7
3838                                <1>     ; (only the overlapped main memory pages will be allocated)
3839                                <1>     ; 28/11/2023
3840                                <1>     ; (but all LFB pages will be added to the kernel's page tables)
3841                                <1>     ; (NOTE: LFB size -which will be used- is the size of the max.
3842                                <1>     ;  recognized screen resolution. For example: 1024*768*4 bytes)
3843                                <1>     ; 15/12/2020
3844                                <1>     ; 14/12/2020 - TRDOS 386 v2.0.3
3845                                <1>     ; Set kernel page tables for linear frame buffer
3846                                <1>     ;
3847                                <1>     ; Input:
3848                                <1>     ;       ;[LFB_ADDR] = linear frame buffer base address
3849                                <1>     ;       ;[LFB_SIZE] = linear frame buffer size in bytes
3850                                <1>     ;       ; 29/11/2023
3851                                <1>     ;       ;ebx = Linear frame buffer base address as page num
3852                                <1>     ;       ; 20/10/2023
3853                                <1>     ;       esi = linear frame buffer base address
3854                                <1>     ;       ecx = memory size in pages = [memory_size]
3855                                <1>     ;       ; 28/11/2023
3856                                <1>     ;       edx = linear frame buffer size in pages
3857                                <1>     ; Output:
3858                                <1>     ;       none
3859                                <1>     ;       cf = 1 -> error
3860                                <1>     ;
3861                                <1>     ; Modified registers: eax, ebx, ecx, edx, esi, edi, ebp
3862                                <1>
3863                                <1>
3864                                <1> ; 02/12/2023 ; (!**!)
3865                                <1> ; ('memory size' -for M.A.T.- is always <= LFB stat/base address)
3866                                <1> %if 0
3867                                <1>
3868                                <1>     ; 29/11/2023
3869                                <1>     ; allocate LFB blocks which < memory size at first
3870                                <1>
3871                                <1>     ;mov    eax, edx
3872                                <1>     ;shl    eax, 12
3873                                <1>     ;add    eax, esi
3874                                <1>     ;jc     short a_lfb_k__err ; > 4GB limit
3875                                <1>     ;
3876                                <1>     ;;mov   eax, 1048576 ; page count of (full) 4GB memory
3877                                <1>     ;;sub   eax, edx
3878                                <1>     ;;cmp   eax, esi
3879                                <1>     ;jb     short a_lfb_k__err ; > 4GB limit
3880                                <1>
3881                                <1>     mov     ebx, esi ; LFB base/start address
3882                                <1>     shr     ebx, 12  ; convert byte address to page address
3883                                <1>     ; ebx = LFB start/base page number
3884                                <1>
3885                                <1>     sub     ecx, ebx
3886                                <1>     jna     short a_lfb_k_2 ; skip M.A.T. bit allocation
3887                                <1>     cmp     ecx, edx
3888                                <1>     jna     short a_lfb_k_0
3889                                <1>     mov     ecx, edx
3890                                <1> a_lfb_k_0:
3891                                <1>     sub     edx, ecx
3892                                <1>     jna     short a_lfb_k_1
3893                                <1>
3894                                <1>     ; edx = remain count of LFB pages which are out of the M.A.T.
3895                                <1>     ; ebx = LFB start/base page number
3896                                <1>     ; ecx = page allocation count in the M.A.T.
3897                                <1>
3898                                <1>     ;mov    eax, 4096
3899                                <1>     ;mul    ecx
3900                                <1>     mov     eax, ecx
3901                                <1>     shl     eax, 12 ; * 4096
3902                                <1>
3903                                <1>     add     esi, eax
3904                                <1>     ; esi = LFB address just after the main memory if there is
3905                                <1> a_lfb_k_1:
3906                                <1>     mov     eax, ebx ; LFB start/base page number
3907                                <1>     call    allocate_lfb_memory_block
3908                                <1>
3909                                <1>     ; here if there is, remain (the 2nd) part of the LFB
3910                                <1>     ; will be allocated by using new kernel page table(s)
3911                                <1>     ; (but M.A.T. bits will not be cleared because
3912                                <1>     ;  the 2nd LFB part is out of the M.A.T.)
3913                                <1>
3914                                <1>     and     edx, edx
3915                                <1>     jz      short a_lfb_k_6  ; all LFB pages have been allocated
3916                                <1>
3917                                <1> %endif
3918                                <1>
3919                                <1> a_lfb_k_2:
3920                                <1>     ; 29/11/2023
3921 0000627F C1EE0C                <1>     shr     esi, 12 ; convert LFB address to page number
3922                                <1>     ; edx = count of pages to be added (in kernel page tables)
3923                                <1>     ; 28/11/2023
3924 00006282 89D5                 <1>     mov     ebp, edx
3925 00006284 81C2FF030000         <1>     add     edx, 1023 ; round up
3926 0000628A C1EA0A               <1>     shr     edx, 10  ; / 1024
3927                                <1>     ; edx = count of kernel page tables to be added
3928                                <1>     ; (edx = 1 or edx = 2)
3929                                <1>     ; ((1024*768*4 = 3145728 bytes, 768 PTEs, 1 page table))
3930                                <1>     ; ((1920*1080*4 = 8294400 bytes, 2025 PTEs, 2 page tables))
3931 0000628D E894F5FFFF             <1>     call    allocate_page
3932                                <1>     ;jc     short a_lfb_k__err
3933                                <1>     ; (mem alloc error is not expected at this startup stage)
3934                                <1>     ; eax = physical address of the new page table
3935 00006292 89F7                 <1>     mov     edi, esi  ; LFB start/base page number
3936 00006294 C1EF0A               <1>     shr     edi, 10
3937                                <1>     ; edi = PDE entry number of the Linear Frame Buffer addr
3938                                <1>     ; (edi = 832 for 0D0000000h, edi = 896 for 0E0000000h)
3939 00006297 C1E702               <1>     shl     edi, 2
3940                                <1>     ; edi = PDE offset
3941 0000629A 033D[80760100]       <1>     add     edi, [k_page_dir] ; Kernel's Page Dir Address
3942 000062A0 50                   <1>     push    eax ; +
3943 000062A1 660D0304             <1>     or      ax, PDE_A_PRESENT + PDE_A_WRITE + PDE_EXTERNAL
3944                                <1>                             ; supervisor + read&write + present
3945                                <1>                             ; + external memory block (LFB)
3946 000062A5 AB                   <1>     stosd
3947                                <1>
3948 000062A6 4A                   <1>     dec     edx
3949 000062A7 740C                 <1>     jz      short a_lfb_k_3       ; only 1 new page table
3950                                <1>
3951                                <1>     ; the 2nd page table
3952 000062A9 E878F5FFFF           <1>     call    allocate_page
3953                                <1>     ;jc     short a_lfb_k__err
3954                                <1>     ; (mem alloc error is not expected at this startup stage)
```

```
                                <1>     ; eax = physical address of the new page table
000062AE 89C3                   <1>     mov     ebx, eax
000062B0 660D0304               <1>     or      ax, PDE_A_PRESENT + PDE_A_WRITE + PDE_EXTERNAL
                                <1>                             ; supervisor + read&write + present
                                <1>                             ; + external memory block (LFB)
000062B4 AB                     <1>     stosd
                                <1>
                                <1> a_lfb_k_3:
                                <1>     ; set new PTEs
                                <1>     ;
                                <1>     ; ebp = count of pages (PTEs) to be allocated
000062B5 5F                     <1>     pop     edi ; +  ; 1st page table address
                                <1>     ; 29/11/2023
000062B6 B900040000             <1>     mov     ecx, 1024       ; number of PTEs in a page table
000062BB 89F0                   <1>     mov     eax, esi        ; LFB base/start page number
                                <1>                             ; (may be the 2nd part of the LFB)
000062BD C1E00C                 <1>     shl     eax, 12
000062C0 660D0304               <1>     or      ax, PTE_A_PRESENT + PTE_A_WRITE + PTE_EXTERNAL
                                <1>                             ; supervisor + read&write + present
                                <1>                             ; + external memory block (LFB)
                                <1> a_lfb_k_4:
000062C4 AB                     <1>     stosd                   ; save as PTE
000062C5 4D                     <1>     dec     ebp
000062C6 740D                   <1>     jz      short a_lfb_k_5
000062C8 0500100000             <1>     add     eax, 4096       ; the next page address
000062CD E2F5                   <1>     loop    a_lfb_k_4
                                <1>
                                <1>     ;and    ebx, ebx
                                <1>     ;jz     short a_lfb_k_6
000062CF 89DF                   <1>     mov     edi, ebx        ; the 2nd page table address
                                <1>     ;xor    ebx, ebx
                                <1>     ;mov    ecx, 1024
000062D1 B504                   <1>     mov     ch, 4 ; cx = 4*256 = 1024
000062D3 EBEF                   <1>     jmp     short a_lfb_k_4 ; only 1 more loop (no the 3rd pt)
                                <1>                             ; (ebp <= 1024 here)
                                <1> a_lfb_k_5:
000062D5 49                     <1>     dec     ecx
000062D6 7404                   <1>     jz      short a_lfb_k_6
000062D8 31C0                   <1>     xor     eax, eax        ; clear page table entry (empty)
000062DA F3AB                   <1>     rep     stosd
                                <1> a_lfb_k_6:
                                <1>     ; 29/11/2023
000062DC C3                     <1>     retn
                                <1>
                                <1> ; 28/11/2023
                                <1> %if 0
                                <1>
                                <1> allocate_lfb_pages_for_kernel:
                                <1>     ; 20/10/2023 - TRDOS 386 v2.0.7
                                <1>     ; (only the overlapped main memory pages will be allocated)
                                <1>     ; ((if LFB base/start address is greater than -or equal to-
                                <1>     ;       main memory size, this procedure is not called/used))
                                <1>     ; 15/12/2020
                                <1>     ; 14/12/2020 - TRDOS 386 v2.0.3
                                <1>     ; Set kernel page tables for linear frame buffer
                                <1>     ;
                                <1>     ; Input:
                                <1>     ;       ;[LFB_ADDR] = linear frame buffer base address
                                <1>     ;       ;[LFB_SIZE] = linear frame buffer size in bytes
                                <1>     ;       ; 20/10/2023
                                <1>     ;       eax = Linear frame buffer base address as page num
                                <1>     ;             (eax < memory size)
                                <1>     ;       esi = linear frame buffer base address
                                <1>     ;       ecx = memory size in pages = [memory_size]
                                <1>     ; Output:
                                <1>     ;       none
                                <1>     ;       cf = 1 -> error
                                <1>     ;
                                <1>     ; Modified registers: eax, ecx, edx, edi
                                <1>
                                <1>     ; 20/10/2023
                                <1>     ;mov    edi, [LFB_ADDR]
                                <1>     ;mov    edx, [LFB_SIZE]
                                <1>
                                <1>     ;;;
                                <1>     ; 20/10/2023
                                <1>     mov     edi, eax ; LFB base/start page number
                                <1>     ;mov    ecx, [memory_size]
                                <1>     sub     ecx, eax
                                <1>
                                <1>     ; ecx = number of pages to be set as (already) allocated
                                <1>     ;;;
                                <1>
                                <1>     ; 20/10/2023
                                <1>     shr     edi, 10 ; convert page number to PDE entry number
                                <1>                     ; (1024 pages per page table -the shift
                                <1>                     ; result is a page table index number
                                <1>                     ; or page directory entry number <= 1023)
                                <1>     ;shr    edi, 22 ; convert address to page number
                                <1>                     ; and then convert it to PDE entry offset
                                <1>                     ; (1 PDE is for 4MB, 22 bit shift)
                                <1>
                                <1>     shl     di, 2   ; * 4 to obtain the PDE offset
                                <1>
                                <1>     ; 20/10/2023
                                <1>     ;;add   edx, 4095
                                <1>     ;shr    edx, 12 ; convert LFB size to LFB page count
                                <1>
                                <1>     ;mov    ecx, edx ; * ; LFB page count
                                <1>     ; 20/10/2023
                                <1>     mov     edx, ecx ; *
                                <1>                     ; 20/10/2023
                                <1>                     ; edx = page count (to be set as allocated)
                                <1>                     ;       from the LFB start page
                                <1>                     ; (it may be <= linear frame buffer size)
                                <1>
                                <1>     add     ecx, 1023 ; page count + 1023
                                <1>     shr     ecx, 10 ; convert to page directory entry count
                                <1>                     ; (page table count)
                                <1>     push    ecx ; **
                                <1>     shl     ecx, 12 ; convert to byte count
                                <1>
                                <1>     ; 20/10/2023
                                <1>     ; (absolute allocation is needed instead of the first fit)
                                <1>     ;xor    eax, eax ; first available pages
                                <1>
                                <1>     ; 20/10/2023
                                <1>     ;mov    esi, [LFB_ADDR]
                                <1>     mov     eax, esi ; linear frame buffer's base/start address
                                <1>
                                <1>     ; allocate contiguous memory block for these kernel pages
                                <1>
                                <1>     ;call   allocate_memory_block
                                <1>     ; 20/10/2023
```

```
4079                            <1>     ; (short way to set LFB page bits on the 'M.A.T.)
4080                            <1>     call    allocate_lfb_memory_block ; in 'allocate_memory_block'
4081                            <1>
4082                            <1>     ; eax = start address of (contiguous) memory block
4083                            <1>     pop     ecx ; ** ; PDE count
4084                            <1>     jnc     short a_lfb_k_1
4085                            <1>     ; error (cf=1)
4086                            <1>     retn
4087                            <1> a_lfb_k_1:
4088                            <1>     ; Allocate (new) page tables in kernel's page directory
4089                            <1>     push    ecx ; PDE (page table) count
4090                            <1>     push    eax ; start address of contiguous memory pages
4091                            <1>                 ; (at page boundary)
4092                            <1>     ; edi = 1st page directory entry offset
4093                            <1>     add     edi, [k_page_dir] ; Kernel's Page Dir Address
4094                            <1> a_lfb_k_2:
4095                            <1>     or      ax, PDE_A_PRESENT + PDE_A_WRITE + PDE_EXTERNAL
4096                            <1>                         ; supervisor + read&write + present
4097                            <1>                         ; + external memory block (LFB)
4098                            <1>     stosd
4099                            <1>     add     eax, 4096
4100                            <1>     loop    a_lfb_k_2
4101                            <1>
4102                            <1>     pop     edi ; start addr of contiguous memory pages
4103                            <1>     pop     ecx ; page table (PDE) count
4104                            <1>
4105                            <1>     ; Allocate pages in (new) kernel page tables
4106                            <1>
4107                            <1>     ; (Note: page tables are contiguous in pyhsical memory)
4108                            <1>     shl     ecx, 10 ; * 1024, convert to (total) PTE count
4109                            <1>
4110                            <1>     mov     eax, [LFB_ADDR]
4111                            <1>     ; edx = LFB page count
4112                            <1>     ;and    ax, ~4095  ; lw of LFB address is 0
4113                            <1> a_lfb_k_3:
4114                            <1>     or      ax, PTE_A_PRESENT + PTE_A_WRITE + PTE_EXTERNAL
4115                            <1>                         ; supervisor + read&write + present
4116                            <1>                         ; + external memory block (LFB)
4117                            <1>     stosd
4118                            <1>     dec     edx
4119                            <1>     jz      short a_lfb_k_4 ; LFB size has been completed (!?)
4120                            <1>     add     eax, 4096
4121                            <1>     loop    a_lfb_k_3
4122                            <1>
4123                            <1>     retn
4124                            <1>
4125                            <1> a_lfb_k_4:
4126                            <1>     ; clear PTEs for empty/free pages
4127                            <1>     ;       (if there are after LFB !?)
4128                            <1>     xor     eax, eax ; clear page table entry (empty)
4129                            <1>     rep     stosd
4130                            <1>     retn
4131                            <1>
4132                            <1> %endif
4133                            <1>
4134                            <1> ;deallocate_lfb_pages_for_kernel:
4135                            <1>     ; 15/12/2020
4136                            <1>     ; 14/12/2020 - TRDOS 386 v2.0.3
4137                            <1>     ; Reset/Release kernel page tables
4138                            <1>     ;       which are used for linear frame buffer
4139                            <1>     ; (this procedure will be called by kernel only)
4140                            <1>     ;
4141                            <1>     ; Input:
4142                            <1>     ;       [LFB_ADDR] = linear frame buffer base address
4143                            <1>     ;       [FFB_SIZE] = linear frame buffer size in bytes
4144                            <1>     ; Output:
4145                            <1>     ;       none
4146                            <1>     ;
4147                            <1>     ; Modified registers: eax, ecx, edi
4148                            <1>
4149                            <1>     ;mov    edi, [LFB_ADDR]
4150                            <1>     ;mov    ecx, [LFB_SIZE]
4151                            <1>     ;
4152                            <1>     ;shr    edi, 22 ; convert address to page number
4153                            <1>     ;               ; and then convert it to PDE entry offset
4154                            <1>     ;               ; (1 PDE is for 4MB, 22 bit shift)
4155                            <1>     ;
4156                            <1>     ;shl    di, 2   ; * 4 for offset
4157                            <1>     ;
4158                            <1>     ;;add   ecx, 4095
4159                            <1>     ;shr    ecx, 12 ; convert LFB size to page count
4160                            <1>     ;
4161                            <1>     ;add    ecx, 1023 ; page count + 1023
4162                            <1>     ;shr    ecx, 10 ; convert to page directory entry count
4163                            <1>     ;               ; (page table count)
4164                            <1>     ;push   ecx ; *
4165                            <1>     ;shl    ecx, 12 ; convert to byte count
4166                            <1>     ;
4167                            <1>     ;xor    eax, eax ; first available pages
4168                            <1>     ;
4169                            <1>     ;; deallocate contiguous memory block for kernel pages
4170                            <1>     ;
4171                            <1>     ;call   deallocate_memory_block
4172                            <1>     ;
4173                            <1>     ;pop    ecx ; * ; PDE count
4174                            <1>     ;
4175                            <1>     ;; Release/Free PDEs (page tables) in kernel's page dir
4176                            <1>     ;; edi = 1st page directory entry offset
4177                            <1>     ;add    edi, [k_page_dir] ; Kernel's Page Dir Address
4178                            <1>     ;sub    eax, eax ; clear (also invalidate)
4179                            <1>     ;rep    stosd
4180                            <1>     ;
4181                            <1>     ;retn
4182                            <1>
4183                            <1> ; /// End Of MEMORY MANAGEMENT FUNCTIONS ///
4184                            <1>
4185                            <1> ;; Data:
4186                            <1>
4187                            <1> ; 09/03/2015
4188                            <1> ;swpq_count: dw 0 ; count of pages on the swap que
4189                            <1> ;swp_drv:    dd 0 ; logical drive description table address of the swap drive/disk
4190                            <1> ;swpd_size:  dd 0 ; size of swap drive/disk (volume) in sectors (512 bytes).
4191                            <1> ;swpd_free:  dd 0 ; free page blocks (4096 bytes) on swap disk/drive (logical)
4192                            <1> ;swpd_next:  dd 0 ; next free page block
4193                            <1> ;swpd_last:  dd 0 ; last swap page block
3229                                %include 'timer.s'   ; 17/01/2015
   1                            <1> ; ****************************************************************************
   2                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.5 - timer.s
   3                            <1> ; ----------------------------------------------------------------------------
   4                            <1> ; Last Update: 08/08/2022  (Previous: 18/04/2021)
   5                            <1> ; ----------------------------------------------------------------------------
   6                            <1> ; Beginning: 17/01/2016
   7                            <1> ; ----------------------------------------------------------------------------
   8                            <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> ;
                        <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> ;
                        <1> ; ///////// TIMER (& REAL TIME CLOCK) FUNCTIONS ////////////////
                        <1>
                        <1> int1Ah:
                        <1>     ; 29/01/2016
                        <1>     ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
000062DD 9C             <1>     pushfd
000062DE 0E             <1>     push    cs
000062DF E801000000     <1>     call    TIME_OF_DAY_1
000062E4 C3             <1>     retn
                        <1>
                        <1> ;--------------------------------------------------------------------------------
                        <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> ; 29/07/2022
                        <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>     ; 07/08/5022
                        <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
                        <1>     ;sti                            ; INTERRUPTS BACK ON
                        <1>     ; 29/05/2016
000062E5 80642408FE     <1>     and     byte [esp+8], 11111110b      ; clear carry bit of eflags register
                        <1>     ;
000062EA 80FC08         <1>     cmp     ah, (RTC_TBE-RTC_TB)/4; CHECK IF COMMAND IN VALID RANGE (0-7)
000062ED F5             <1>     cmc                             ; COMPLEMENT CARRY FOR ERROR EXIT
                        <1>     ; (*) jc short TIME_9           ; EXIT WITH CARRY = 1 IF NOT VALID
000062EE 721A           <1>     jc      short _TIME_9 ; 29/05/2016
                        <1>
000062F0 1E             <1>     push    ds
000062F1 56             <1>     push    esi
000062F2 66BE1000       <1>     mov     si, KDATA               ; kernel data segment
000062F6 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>     ;
000062F8 C0E402             <1>     shl     ah, 2                 ; convert function to dword offset
000062FB 0FB6F4             <1>     movzx   esi, ah               ; PLACE INTO ADDRESSING REGISTER
                            <1>     ;cli                          ; NO INTERRUPTS DURING TIME FUNCTIONS
000062FE FF96[10630000]     <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
00006304 B400               <1>     mov     ah, 0                 ; CLEAR (AH) TO ZERO
00006306 5E                 <1>     pop     esi                   ; RECOVER USERS REGISTER
00006307 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
00006308 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
0000630A 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:
0000630F 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:
00006310 [30630000]         <1>     dd      RTC_00                ; 0 = READ CURRENT CLOCK COUNT
00006314 [43630000]         <1>     dd      RTC_10                ; 1 = SET CLOCK COUNT
00006318 [51630000]         <1>     dd      RTC_20                ; 2 = READ THE REAL TIME CLOCK TIME
0000631C [7F630000]         <1>     dd      RTC_30                ; 3 = SET REAL TIME CLOCK TIME
00006320 [BC630000]         <1>     dd      RTC_40                ; 4 = READ THE REAL TIME CLOCK DATE
00006324 [E2630000]         <1>     dd      RTC_50                ; 5 = SET REAL TIME CLOCK DATE
00006328 [41640000]         <1>     dd      RTC_60                ; 6 = SET THE REAL TIME CLOCK ALARM
0000632C [93640000]         <1>     dd      RTC_70                ; 7 = RESET ALARM
                            <1>
                            <1> RTC_TBE     equ     $
                            <1>
                            <1> RTC_00:                           ; READ TIME COUNT
00006330 A0[04770100]       <1>     mov     al, [TIMER_OFL]             ; GET THE OVERFLOW FLAG
00006335 C605[04770100]00   <1>     mov     byte [TIMER_OFL], 0   ; AND THEN RESET THE OVERFLOW FLAG
0000633C 8B0D[00770100]     <1>         mov     ecx, [TIMER_LH]         ; GET COUNT OF TIME
00006342 C3                 <1>     retn
                            <1>
                            <1> RTC_10:                           ; SET TIME COUNT
00006343 890D[00770100]     <1>         mov     [TIMER_LH], ecx         ; SET TIME COUNT
00006349 C605[04770100]00   <1>     mov     byte [TIMER_OFL], 0   ; RESET OVERFLOW FLAG
00006350 C3                 <1>     retn                          ; RETURN WITH NO CARRY
                            <1>
                            <1> RTC_20:                           ; GET RTC TIME
00006351 E8C7010000         <1>     call    UPD_IPR               ; CHECK FOR UPDATE IN PROCESS
00006356 7226               <1>     jc      short RTC_29          ; EXIT IF ERROR (CY= 1)
                            <1>
00006358 B000               <1>     mov     al, CMOS_SECONDS      ; SET ADDRESS OF SECONDS
0000635A E8F4010000         <1>     call    CMOS_READ             ; GET SECONDS
0000635F 88C6               <1>     mov     dh, al                ; SAVE
00006361 B00B               <1>     mov     al, CMOS_REG_B        ; ADDRESS ALARM REGISTER
00006363 E8EB010000         <1>     call    CMOS_READ             ; READ CURRENT VALUE OF DSE BIT
00006368 2401               <1>     and     al, 00000001b         ; MASK FOR VALID DSE BIT
0000636A 88C2               <1>     mov     dl, al                ; SET [DL] TO ZERO FOR NO DSE BIT
0000636C B002               <1>     mov     al, CMOS_MINUTES      ; SET ADDRESS OF MINUTES
0000636E E8E0010000         <1>     call    CMOS_READ             ; GET MINUTES
00006373 88C1               <1>     mov     cl, al                ; SAVE
00006375 B004               <1>         mov     al, CMOS_HOURS          ; SET ADDRESS OF HOURS
                            <1> RTC_41:             ; 29/07/2022
00006377 E8D7010000         <1>     call    CMOS_READ             ; GET HOURS
0000637C 88C5               <1>     mov     ch, al                ; SAVE
                            <1>     ; 29/07/2022
                            <1>     ;clc                          ; SET CY= 0
                            <1> RTC_29:
0000637E C3                 <1>     retn                          ; RETURN WITH RESULT IN CARRY FLAG
                            <1>
                            <1> RTC_30:                           ; SET RTC TIME
0000637F E899010000         <1>     call    UPD_IPR               ; CHECK FOR UPDATE IN PROCESS
00006384 7305               <1>     jnc     short RTC_35          ; GO AROUND IF CLOCK OPERATING
00006386 E8AD010000         <1>     call    RTC_STA               ; ELSE TRY INITIALIZING CLOCK
                            <1> RTC_35:
0000638B 88F4               <1>     mov     ah, dh                ; GET TIME BYTE - SECONDS
0000638D B000               <1>     mov     al, CMOS_SECONDS      ; ADDRESS SECONDS
0000638F E894000000         <1>     call    CMOS_WRITE            ; UPDATE SECONDS
00006394 88CC               <1>     mov     ah, cl                ; GET TIME BYTE - MINUTES
00006396 B002               <1>     mov     al, CMOS_MINUTES      ; ADDRESS MINUTES
00006398 E88B000000         <1>     call    CMOS_WRITE            ; UPDATE MINUTES
0000639D 88EC               <1>     mov     ah, ch                ; GET TIME BYTE - HOURS
0000639F B004               <1>     mov     al, CMOS_HOURS        ; ADDRESS HOURS
000063A1 E882000000         <1>     call    CMOS_WRITE            ; UPDATE ADDRESS
                            <1>     ;mov    al, CMOS_REG_B        ; ADDRESS ALARM REGISTER
                            <1>     ;mov    ah, al
000063A6 66B80B0B           <1>     mov     ax, CMOS_REG_B * 257
000063AA E8A4010000         <1>     call    CMOS_READ             ; READ CURRENT TIME
000063AF 2462               <1>     and     al, 01100010b         ; MASK FOR VALID BIT POSITIONS
000063B1 0C02               <1>     or      al, 00000010b         ; TURN ON 24 HOUR MODE
000063B3 80E201             <1>     and     dl, 00000001b         ; USE ONLY THE DSE BIT
000063B6 08D0               <1>     or      al, dl                ; GET DAY LIGHT SAVINGS TIME BIT (OSE)
000063B8 86C4               <1>     xchg    ah, al                ; PLACE IN WORK REGISTER AND GET ADDRESS
                            <1>     ;call   CMOS_WRITE            ; SET NEW ALARM SITS
                            <1>     ;clc                          ; SET CY= 0
                            <1>     ;retn                         ; RETURN WITH CY= 0
```

```
                        <1>     ; 29/07/2022
000063BA EB6C           <1>     jmp     short CMOS_WRITE
                        <1>
                        <1> RTC_40:                             ; GET RTC DATE
000063BC E85C010000     <1>     call    UPD_IPR                 ; CHECK FOR UPDATE IN PROCESS
                        <1>     ;jc     short RTC_49            ; EXIT IF ERROR (CY= 1)
                        <1>     ; 07/08/2022
000063C1 72BB           <1>     jc      short RTC_29
                        <1>
000063C3 B007           <1>     mov     al, CMOS_DAY_MONTH      ; ADDRESS DAY OF MONTH
000063C5 E889010000     <1>     call    CMOS_READ               ; READ DAY OF MONTH
000063CA 88C2           <1>     mov     dl, al                  ; SAVE
000063CC B008           <1>     mov     al, CMOS_MONTH          ; ADDRESS MONTH
000063CE E880010000     <1>     call    CMOS_READ               ; READ MONTH
000063D3 88C6           <1>     mov     dh, al                  ; SAVE
000063D5 B009           <1>     mov     al, CMOS_YEAR           ; ADDRESS YEAR
000063D7 E877010000     <1>     call    CMOS_READ               ; READ YEAR
000063DC 88C1           <1>     mov     cl, al                  ; SAVE
000063DE B032           <1>     mov     al, CMOS_CENTURY        ; ADDRESS CENTURY LOCATION
                        <1> ; 29/07/2022
                        <1> ;   call    CMOS_READ               ; GET CENTURY BYTE
                        <1> ;   mov     ch, al                  ; SAVE
                        <1> ;   ; 29/07/2022
                        <1> ;   ;clc                            ; SET CY=0
                        <1> ;RTC_49:
                        <1> ;   retn                            ; RETURN WITH RESULTS IN CARRY FLAG
                        <1>
                        <1>     ; 29/07/2022
000063E0 EB95           <1>     jmp     short RTC_41
                        <1>
                        <1>
                        <1> RTC_50:                             ; SET RTC DATE
000063E2 E836010000     <1>     call    UPD_IPR                 ; CHECK FOR UPDATE IN PROCESS
000063E7 7305           <1>     jnc     short RTC_55            ; GO AROUND IF NO ERROR
000063E9 E84A010000     <1>     call    RTC_STA                 ; ELSE INITIALIZE CLOCK
                        <1> RTC_55:
000063EE 66B80600       <1>     mov     ax, CMOS_DAY_WEEK       ; ADDRESS OF DAY OF WEEK BYTE
000063F2 E831000000     <1>     call    CMOS_WRITE              ; LOAD ZEROS TO DAY OF WEEK
000063F7 88D4           <1>     mov     ah, dl                  ; GET DAY OF MONTH BYTE
000063F9 B007           <1>     mov     al, CMOS_DAY_MONTH      ; ADDRESS DAY OF MONTH BYTE
000063FB E828000000     <1>     call    CMOS_WRITE              ; WRITE OF DAY OF MONTH REGISTER
00006400 88F4           <1>     mov     ah, dh                  ; GET MONTH
00006402 B008           <1>     mov     al, CMOS_MONTH          ; ADDRESS MONTH BYTE
00006404 E81F000000     <1>     call    CMOS_WRITE              ; WRITE MONTH REGISTER
00006409 88CC           <1>     mov     ah, cl                  ; GET YEAR BYTE
0000640B B009           <1>     mov     al, CMOS_YEAR           ; ADDRESS YEAR REGISTER
0000640D E816000000     <1>     call    CMOS_WRITE              ; WRITE YEAR REGISTER
00006412 88EC           <1>     mov     ah, ch                  ; GET CENTURY BYTE
00006414 B032           <1>     mov     al, CMOS_CENTURY        ; ADDRESS CENTURY BYTE
00006416 E80D000000     <1>     call    CMOS_WRITE              ; WRITE CENTURY LOCATION
                        <1>     ;mov    al, CMOS_REG_B          ; ADDRESS ALARM REGISTER
                        <1>     ;mov    ah, al
0000641B 66B80B0B       <1>     mov     ax, CMOS_REG_B * 257
0000641F E82F010000     <1>     call    CMOS_READ               ; READ CURRENT SETTINGS
00006424 247F           <1>     and     al, 07Fh                ; CLEAR 'SET BIT'
00006426 86C4           <1>     xchg    ah, al                  ; MOVE TO WORK REGISTER
                        <1>     ;call   CMOS_WRITE              ; AND START CLOCK UPDATING
                        <1>     ;clc                            ; SET CY= 0
                        <1>     ;retn                           ; RETURN CY=0
                        <1>     ; 29/07/2022
                        <1>     ;jmp    short CMOS_WRITE
                        <1>
                        <1> ;---------------------------------------------------------------------------
                        <1>
                        <1> ; 08/08/2022
                        <1> ; 29/07/2022 (TRDOS 386 v2.0.5)
                        <1> ; 18/04/2021 (TRDOS 386 v2.0.4)
                        <1> ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>
                        <1> ;--- CMOS_WRITE -----------------------------------------------------------
                        <1> ;   WRITE BYTE TO CMOS SYSTEM CLOCK CONFIGURATION TABLE              :
                        <1> ;                                                                    :
                        <1> ; INPUT: (AL)=     CMOS TABLE ADDRESS TO BE WRITTEN TO                    :
                        <1> ;          BIT    7 = 0 FOR NMI ENABLED AND 1 FOR NMI DISABLED ON EXIT    :
                        <1> ;          BITS 6-0 = ADDRESS OF TABLE LOCATION TO WRITE                  :
                        <1> ;    (AH)= NEW VALUE TO BE PLACED IN THE ADDRESSED TABLE LOCATION          :
                        <1> ;                                                                    :
                        <1> ; OUTPUT:   VALUE IN (AH) PLACED IN LOCATION (AL) WITH NMI LEFT DISABLED   :
                        <1> ;           IF BIT 7 OF (AL) IS ON, DURING THE CMOS UPDATE BOTH NMI AND    :
                        <1> ;           NORMAL INTERRUPTS ARE DISABLED TO PROTECT CMOS DATA INTEGRITY. :
                        <1> ;           THE CMOS ADDRESS REGISTER IS POINTED TO A DEFAULT VALUE AND    :
                        <1> ;           THE INTERRUPT FLAG RESTORED TO THE ENTRY STATE ON RETURN.       :
                        <1> ;           ONLY THE CMOS LOCATION AND THE NMI STATE IS CHANGED.            :
                        <1> ;---------------------------------------------------------------------------
                        <1>
                        <1>     ; 08/08/2022
                        <1> CMOS_WRITE:                         ; WRITE (AH) TO LOCATION (AL)
                        <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
                        <1>     ;pushf                          ; SAVE INTERRUPT ENABLE STATUS AND FLAGS
                        <1>     ;;push  ax                      ; SAVE WORK REGISTER VALUES
                        <1>     ; 18/04/2021
                        <1>     ;push   eax
00006428 D0C0           <1>     rol     al, 1                   ; MOVE NMI BIT TO LOW POSITION
0000642A F9             <1>     stc                             ; FORCE NMI BIT ON IN CARRY FLAG
0000642B D0D8           <1>     rcr     al, 1                   ; HIGH BIT ON TO DISABLE NMI - OLD IN CY
0000642D FA             <1>     cli                             ; DISABLE INTERRUPTS
0000642E E670           <1>     out     CMOS_PORT, al           ; ADDRESS LOCATION AND DISABLE NMI
00006430 88E0           <1>     mov     al, ah                  ; GET THE DATA BYTE TO WRITE
00006432 E671           <1>     out     CMOS_DATA, al           ; PLACE IN REQUESTED CMOS LOCATION
00006434 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
00006436 D0D8           <1>     rcr     al, 1                   ; PUT ORIGINAL NMI MASK BIT INTO ADDRESS
00006438 E670           <1>     out     CMOS_PORT, al           ; SET DEFAULT TO READ ONLY REGISTER
                        <1>     ;nop                            ; I/O DELAY
                        <1>     ; 29/07/2022
0000643A E6EB           <1>     out     0EBh, al ; NEWIODELAY ; AWARD BIOS 1999, ATIME.ASM
0000643C E471           <1>     in      al, CMOS_DATA           ; OPEN STANDBY LATCH
                        <1>     ;;pop   ax                      ; RESTORE WORK REGISTERS
                        <1>     ; 18/04/2021
                        <1>     ;pop    eax
                        <1>     ;popf
                        <1>     ; 29/07/2022
                        <1>     ;clc
                        <1>     ; 08/08/2022
0000643E 30C0           <1>     xor     al, al
00006440 C3             <1>     retn
                        <1>
                        <1> ;---------------------------------------------------------------------------
                        <1>
                        <1> RTC_60:                             ; SET RTC ALARM
00006441 B00B           <1>     mov     al, CMOS_REG_B          ; ADDRESS ALARM
00006443 E80B010000     <1>     call    CMOS_READ               ; READ ALARM REGISTER
00006448 A820           <1>     test    al, 20h                 ; CHECK FOR ALARM ALREADY ENABLED
0000644A F9             <1>     stc                             ; SET CARRY IN CASE OF ERROR
```

```
                        <1>     jnz     short RTC_69            ; ERROR EXIT IF ALARM SET
                        <1>     call    UPD_IPR                 ; CHECK FOR UPDATE IN PROCESS
                        <1>     jnc     short RTC_65            ; SKIP INITIALIZATION IF NO ERROR
                        <1>     call    RTC_STA                 ; ELSE INITIALIZE CLOCK
                        <1> RTC_65:
                        <1>     mov     ah, dh                  ; GET SECONDS BYTE
                        <1>     mov     al, CMOS_SEC_ALARM      ; ADDRESS THE SECONDS ALARM REGISTER
                        <1>     call    CMOS_WRITE              ; INSERT SECONDS
                        <1>     mov     ah, cl                  ; GET MINUTES PARAMETER
                        <1>     mov     al, CMOS_MIN_ALARM      ; ADDRESS MINUTES ALARM REGISTER
                        <1>     call    CMOS_WRITE              ; INSERT MINUTES
                        <1>     mov     ah, ch                  ; GET HOURS PARAMETER
                        <1>     mov     al, CMOS_HR_ALARM       ; ADDRESS HOUR ALARM REGISTER
                        <1>     call    CMOS_WRITE              ; INSERT HOURS
                        <1>     in      al, INTB01              ; READ SECOND INTERRUPT MASK REGISTER
                        <1>     and     al, 0FEh                ; ENABLE ALARM TIMER BIT (CY= 0)
                        <1>     out     INTB01, al              ; WRITE UPDATED MASK
                        <1>     ;mov    al, CMOS_REG_B          ; ADDRESS ALARM REGISTER
                        <1>     ;mov    ah, al
                        <1>     mov     ax, CMOS_REG_B * 257
                        <1>     call    CMOS_READ               ; READ CURRENT ALARM REGISTER
                        <1>     and     al, 07Fh                ; ENSURE SET BIT TURNED OFF
                        <1>     or      al, 20h                 ; TURN ON ALARM ENABLE
                        <1>     xchg    ah, al                  ; MOVE MASK TO OUTPUT REGISTER
                        <1>     call    CMOS_WRITE              ; WRITE NEW ALARM MASK
                        <1>     ; 29/07/2022
                        <1>     ;clc                            ; SET CY= 0
                        <1> RTC_69:
                        <1>     mov     ax, 0                   ; CLEAR AX REGISTER
                        <1>     retn                            ; RETURN WITH RESULTS IN CARRY FLAC
                        <1>
                        <1> RTC_70:                             ; RESET ALARM
                        <1>     ;mov    al, CMOS_REG_B          ; ADDRESS ALARM REGISTER
                        <1>     ;mov    ah, al
                        <1>     mov     ax, CMOS_REG_B * 257    ; ADDRESS ALARM REGISTER (TO BOTH AH,AL)
                        <1>     call    CMOS_READ               ; READ ALARM REGISTER
                        <1>     and     al, 57h                 ; TURN OFF ALARM ENABLE
                        <1>     xchg    ah, al                  ; SAVE DATA AND RECOVER ADDRESS
                        <1>     ;call   CMOS_WRITE              ; RESTORE NEW VALUE
                        <1>     ;clc                            ; SET CY= 0
                        <1>     ;retn                           ; RETURN WITH NO CARRY
                        <1>     ; 29/07/2022
                        <1>     jmp     short CMOS_WRITE
                        <1>
                        <1> ;--------------------------------------------------------------------------
                        <1>
                        <1> ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>
                        <1> ;--- HARDWARE INT 70 H -- ( IRQ LEVEL 8) ------------------------------------
                        <1> ; ALARM INTERRUPT HANDLER (RTC)                                            :
                        <1> ;       THIS ROUTINE HANDLES THE PERIODIC AND ALARM INTERRUPTS FROM THE CMOS  :
                        <1> ;       TIMER. INPUT FREQUENCY IS 1.024 KHZ OR APPROXIMATELY 1024 INTERRUPTS  :
                        <1> ;       EVERY SECOND FOR THE PERIODIC INTERRUPT. FOR THE ALARM FUNCTION,      :
                        <1> ;       THE INTERRUPT WILL OCCUR AT THE DESIGNATED TIME.                      :
                        <1> ;                                                                              :
                        <1> ;       INTERRUPTS ARE ENABLED WHEN THE EVENT OR ALARM FUNCTION IS ACTIVATED. :
                        <1> ;       FOR THE EVENT INTERRUPT, THE HANDLER WILL DECREMENT THE WAIT COUNTER   :
                        <1> ;       AND WHEN IT EXPIRES WILL SET THE DESIGNATED LOCATION TO 80H. FOR       :
                        <1> ;       THE ALARM INTERRUPT. THE USER MUST PROVIDE A ROUTINE TO INTERCEPT      :
                        <1> ;       THE CORRECT ADDRESS FROM THE VECTOR TABLE INVOKED BY INTERRUPT 4AH     :
                        <1> ;       PRIOR TO SETTING THE REAL TIME CLOCK ALARM (INT 1AH, AH= 06H).         :
                        <1> ;--------------------------------------------------------------------------
                        <1>
                        <1> RTC_A_INT: ; 07/01/2017
                        <1> ;RTC_INT:                           ; ALARM INTERRUPT
                        <1>     push    ds                      ; LEAVE INTERRUPTS DISABLED
                        <1>     push    eax                     ; SAVE REGISTERS
                        <1>     push    edi
                        <1> RTC_I_1:                            ; CHECK FOR SECOND INTERRUPT
                        <1>     mov     ax, 256*(CMOS_REG_B+NMI)+CMOS_REG_C+NMI ; ALARM AND STATUS
                        <1>     out     CMOS_PORT, al           ; WRITE ALARM FLAG MASK ADDRESS
                        <1>     nop                             ; I/O DELAY
                        <1>     jmp     short $+2
                        <1>     in      al, CMOS_DATA           ; READ AND RESET INTERRUPT REQUEST FLAGS
                        <1>     test    al, 01100000b           ; CHECK FOR EITHER INTERRUPT PENDING
                        <1>     jz      short   RTC_I_9         ; EXIT IF NOT A VALID RTC INTERRUPT
                        <1>
                        <1>     xchg    ah, al                  ; SAVE FLAGS AND GET ENABLE ADDRESS
                        <1>     out     CMOS_PORT, al           ; WRITE ALARM ENABLE MASK ADDRESS
                        <1>     nop                             ; I/O DELAY
                        <1>     jmp     short $+2
                        <1>     in      al, CMOS_DATA           ; READ CURRENT ALARM ENABLE MASK
                        <1>     and     al, ah                  ; ALLOW ONLY SOURCES THAT ARE ENABLED
                        <1>     test    al, 01000000b           ; CHECK FOR PERIODIC INTERRUPT
                        <1>     jz      short RTC_I_5           ; SKIP IF NOT A PERIODIC INTERRUPT
                        <1>
                        <1> ;-----      DECREMENT WAIT COUNT BY INTERRUPT INTERVAL
                        <1>
                        <1>     mov     di, KDATA               ; kernel data segment
                        <1>     mov     ds, di
                        <1>
                        <1>     sub     dword [RTC_LH], 976     ; DECREMENT COUNT BY 1/1024
                        <1>
                        <1>     jnc     short RTC_I_5           ; SKIP TILL 32 BIT WORD LESS THAN ZERO
                        <1>
                        <1> ;-----      TURN OFF PERIODIC INTERRUPT ENABLE
                        <1>
                        <1>     ;push   ax                      ; SAVE INTERRUPT FLAG MASK
                        <1>     ; 18/04/2021
                        <1>     push    eax
                        <1>     mov     ax, 257*(CMOS_REG_B+NMI) ; INTERRUPT ENABLE REGISTER
                        <1>     out     CMOS_PORT, al           ; WRITE ADDRESS TO CMOS CLOCK
                        <1>     nop                             ; I/O DELAY
                        <1>     jmp     short $+2
                        <1>     in      al, CMOS_DATA           ; READ CURRENT ENABLES
                        <1>     and     al, 0BFh                ; TURN OFF PIE
                        <1>     xchg    al, ah                  ; GET CMOS ADDRESS AND SAVE VALUE
                        <1>     out     CMOS_PORT, al           ; ADDRESS REGISTER B
                        <1>     xchg    al, ah                  ; GET NEW INTERRUPT ENABLE MASK
                        <1>     out     CMOS_DATA, al           ; SET MASK IN INTERRUPT ENABLE REGISTER
                        <1>     mov     byte [RTC_WAIT_FLAG], 0       ; SET FUNCTION ACTIVE FLAG OFF
                        <1>     mov     edi, [USER_FLAG]        ; SET UP (DS:DI) TO POINT TO USER FLAG
                        <1>     mov     byte [edi], 80h               ; TURN ON USERS FLAG
                        <1>     ;pop    ax                      ; GET INTERRUPT SOURCE BACK
                        <1>     ; 18/04/2021
                        <1>     pop     eax
                        <1> RTC_I_5:
                        <1>     test    al, 00100000b           ; TEST FOR ALARM INTERRUPT
                        <1>     jz      short RTC_I_7           ; SKIP USER INTERRUPT CALL IF NOT ALARM
                        <1>
                        <1>     mov     al, CMOS_REG_D          ; POINT TO DEFAULT READ ONLY REGISTER
                        <1>     out     CMOS_PORT, al           ; ENABLE NMI AND CMOS ADDRESS TO DEFAULT
                        <1>     sti                             ; INTERRUPTS BACK ON NOW
                        <1>     push    edx
```

```
00006506 E8C2B60000          <1>     call    INT4Ah                  ; TRANSFER TO USER ROUTINE
0000650B 5A                  <1>     pop     edx
0000650C FA                  <1>     cli                             ; BLOCK INTERRUPT FOR RETRY
                             <1> RTC_I_7:                            ; RESTART ROUTINE TO HANDLE DELAYED
0000650D EB96                <1>     jmp     short RTC_I_1           ;  ENTRY AND SECOND EVENT BEFORE DONE
                             <1>
                             <1> RTC_I_9:                            ; EXIT - NO PENDING INTERRUPTS
0000650F B00D                <1>     mov     al, CMOS_REG_D          ; POINT TO DEFAULT READ ONLY REGISTER
00006511 E670                <1>     out     CMOS_PORT, al           ; ENABLE NMI AND CMOS ADDRESS TO DEFAULT
00006513 B020                <1>     mov     al, EOI                 ; END OF INTERRUPT MASK TO 8259 - 2
00006515 E6A0                <1>     out     INTB00, al              ; TO 8259 - 2
00006517 E620                <1>     out     INTA00,al               ; TO 8259 - 1
00006519 5F                  <1>     pop     edi                     ; RESTORE REGISTERS
0000651A 58                  <1>     pop     eax
0000651B 1F                  <1>     pop     ds
0000651C CF                  <1>     iretd                           ; END OF INTERRUPT
                             <1>
                             <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
0000651D 51                  <1>     push    ecx
                             <1>
                             <1>     ; 29/05/2016
0000651E 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:
00006523 B00A                <1>     mov     al, CMOS_REG_A          ; ADDRESS STATUS REGISTER A
00006525 FA                  <1>     cli                             ; NO TIMER INTERRUPTS DURING UPDATES
00006526 E828000000          <1>     call    CMOS_READ               ; READ UPDATE IN PROCESS FLAG
0000652B A880                <1>     test    al, 80h                 ; IF UIP BIT IS ON ( CANNOT READ TIME )
0000652D 7406                <1>     jz      short UPD_90            ; EXIT WITH CY= 0 IF CAN READ CLOCK NOW
0000652F FB                  <1>     sti                             ; ALLOW INTERRUPTS WHILE WAITING
00006530 E2F1                <1>     loop    UPD_10                  ; LOOP TILL READY OR TIMEOUT
00006532 31C0                <1>     xor     eax, eax ; xor ax, ax ; CLEAR RESULTS IF ERROR
00006534 F9                  <1>     stc                             ; SET CARRY FOR ERROR
                             <1> UPD_90:
00006535 59                  <1>     pop     ecx                     ; RESTORE CALLERS REGISTER
00006536 FA                  <1>     cli                             ; INTERRUPTS OFF DURING SET
00006537 C3                  <1>     retn                            ; RETURN WITH CY FLAG SET
                             <1>
                             <1>     ; 18/04/2021
                             <1> RTC_STA:                    ; INITIALIZE REAL TIME CLOCK
                             <1>     ;mov    al, CMOS_REG_A          ; ADDRESS REGISTER A AND LOAD DATA MASK
                             <1>     ;mov    ah, 26h
00006538 66B80A26            <1>     mov     ax, (26h*100h)+CMOS_REG_A
0000653C E8E7FEFFFF          <1>     call    CMOS_WRITE              ; INITIALIZE STATUS REGISTER A
                             <1>     ;mov    al, CMOS_REG_B          ; SET "SET BIT" FOR CLOCK INITIALIZATION
                             <1>     ;mov    ah, 82h
00006541 66B80B82            <1>     mov     ax, (82h*100h)+CMOS_REG_B
00006545 E8DEFEFFFF          <1>     call    CMOS_WRITE              ; AND 24 HOUR MODE TO REGISTER B
0000654A B00C                <1>     mov     al, CMOS_REG_C          ; ADDRESS REGISTER C
0000654C E802000000          <1>     call    CMOS_READ               ; READ REGISTER C TO INITIALIZE
00006551 B00D                <1>     mov     al, CMOS_REG_D          ; ADDRESS REGISTER D
                             <1>     ; 18/04/2021
                             <1>     ;call   CMOS_READ               ; READ REGISTER D TO INITIALIZE
                             <1>     ;retn
                             <1>     ;jmp    short CMOS_READ ; 18/04/2021
                             <1>
                             <1> ;---------------------------------------------------------------------------
                             <1>
                             <1>     ; 29/07/2022 - TRDOS 386 v2.0.5
                             <1>     ; 18/04/2021 - TRDOS 386 v2.0.4
                             <1>     ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
                             <1>     ; 22/08/2014 (Retro UNIX 386 v1)
                             <1>     ; IBM PC/AT BIOS source code ----- 10/06/85 (test4.asm)
                             <1>
                             <1> ;--- CMOS_READ ----------------------------------------------------------------
                             <1> ;           READ BYTE FROM CMOS_SYSTEM CLOCK CONFIGURATION TABLE              :
                             <1> ;                                                                             :
                             <1> ; INPUT: (AL)=     CMOS_TABLE ADDRESS TO BE READ                              :
                             <1> ;           BIT    7 = 0 FOR NMI ENABLED AND 1 FOR NMI DISABLED ON EXIT    :
                             <1> ;           BITS 6-0 = ADDRESS OF TABLE LOCATION TO READ                    :
                             <1> ;                                                                           :
                             <1> ; OUTPUT: (AL)     VALUE AT LOCATION (AL) MOVED INTO (AL). IF BIT 7 OF (AL) WAS   :
                             <1> ;           ON THEN NMI LEFT DISABLED, DURING THE CMOS READ BOTH NMI AND   :
                             <1> ;           NORMAL INTERRUPTS ARE DISABLED TO PROTECT CMOS DATA INTEGRITY. :
                             <1> ;           THE CMOS ADDRESS REGISTER IS POINTED TO A DEFAULT VALUE AND    :
                             <1> ;           THE INTERRUPT FLAG RESTORED TO THE ENTRY STATE ON RETURN.       :
                             <1> ;           ONLY THE (AL) REGISTER AND THE NMI STATE IS CHANGED.              :
                             <1> ;---------------------------------------------------------------------------
                             <1>
                             <1> CMOS_READ:
                             <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
                             <1>     ;pushf                          ; SAVE INTERRUPT ENABLE STATUS AND FLAGS
00006553 D0C0                <1>     rol     al, 1                   ; MOVE NMI BIT TO LOW POSITION
00006555 F9                  <1>     stc                             ; FORCE NMI BIT ON IN CARRY FLAG
00006556 D0D8                <1>     rcr     al, 1                   ; HIGH BIT ON TO DISABLE NMI - OLD IN CY
00006558 FA                  <1>     cli                             ; DISABLE INTERRUPTS
00006559 E670                <1>     out     CMOS_PORT, al           ; ADDRESS LOCATION AND DISABLE NMI
                             <1>     ; 29/05/2016
                             <1>     ;nop                            ; I/O DELAY
0000655B E6EB                <1>     out     0EBh, al ; NEWIODELAY ; AWARD BIOS 1999, ATIME.ASM
                             <1>     ;
0000655D E471                <1>     in      al, CMOS_DATA           ; READ THE REQUESTED CMOS LOCATION
                             <1>     ;push   ax      ; SAVE (AH) REGISTER VALUE AND CMOS BYTE
                             <1>     ; 18/04/2021
0000655F 50                  <1>     push    eax
                             <1>     ; 15/03/2015 ; IBM PC/XT Model 286 BIOS source code
                             <1>                  ; ----- 10/06/85 (test4.asm)
00006560 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
00006562 D0D8                <1>     rcr     al, 1                   ; PUT ORIGINAL NMI MASK BIT INTO ADDRESS
00006564 E670                <1>     out     CMOS_PORT, al           ; SET DEFAULT TO READ ONLY REGISTER
                             <1>     ;pop    ax                      ; RESTORE (AH) AND (AL), CMOS BYTE
                             <1>     ; 18/04/2021
00006566 58                  <1>     pop     eax
                             <1>     ; 29/07/2022
                             <1>     ;popf
00006567 F8                  <1>     clc     ; 29/07/2022
00006568 C3                  <1>     retn                            ; RETURN WITH FLAGS RESTORED
                             <1>
                             <1> ;---------------------------------------------------------------------------
                             <1>
                             <1> ; /// End Of TIMER FUNCTIONS ///
                             
00006569 90<rep 7h>              Align 16
```

```
3232
3233                                    gdt:; Global Descriptor Table
3234                                        ; 02/12/2020
3235                                        ; (30/07/2015, conforming cs)
3236                                        ; (26/03/2015)
3237                                        ; (24/03/2015, tss)
3238                                        ; (19/03/2015)
3239                                        ; (29/12/2013)
3240                                        ;
3241 00006570 0000000000000000             dw 0, 0, 0, 0         ; NULL descriptor
3242                                    gdt_kcode:
3243                                        ; 18/08/2014
3244                                                ; 8h kernel code segment, base = 00000000h
3245                                        ;dw 0FFFFh, 0, 9E00h, 00CFh   ; KCODE  ; 30/12/2016
3246 00006578 FFFF0000009ACF00             dw 0FFFFh, 0, 9A00h, 00CFh    ; KCODE
3247                                    gdt_kdata:
3248                                                ; 10h kernel data segment, base = 00000000h
3249 00006580 FFFF00000092CF00             dw 0FFFFh, 0, 9200h, 00CFh    ; KDATA
3250                                    gdt_ucode:
3251                                                ; 1Bh user code segment, base address = 400000h ; CORE
3252                                        ;dw 0FBFFh, 0, 0FE40h, 00CFh  ; UCODE  ; 30/12/2016
3253 00006588 FFFB000040FACF00             dw 0FBFFh, 0, 0FA40h, 00CFh   ; UCODE
3254                                    gdt_udata:
3255                                                ; 23h user data segment, base address = 400000h ; CORE
3256 00006590 FFFB000040F2CF00             dw 0FBFFh, 0, 0F240h, 00CFh   ; UDATA
3257                                    gdt_tss:
3258                                                ; Task State Segment
3259 00006598 6700                         dw 0067h ; Limit = 103 ; (104-1, tss size = 104 byte,
3260                                                                ;  no IO permission in ring 3)
3261                                    gdt_tss0:
3262 0000659A 0000                         dw 0  ; TSS base address, bits 0-15
3263                                    gdt_tss1:
3264 0000659C 00                           db 0  ; TSS base address, bits 16-23
3265                                                ; 49h
3266 0000659D E9                           db 11101001b ; 0E9h => P=1/DPL=11/0/1/0/B/1 --> B = Task is busy (1)
3267 0000659E 00                           db 0 ; G/0/0/AVL/LIMIT=0000 ; (Limit bits 16-19 = 0000) (G=0, 1 byte)
3268                                    gdt_tss2:
3269 0000659F 00                           db 0  ; TSS base address, bits 24-31
3270
3271                                        ; 04/12/2023 - TRDOS v2.0.7
3272                                        ; 30/11/2020
3273                                        ; 29/11/2020 - TRDOS v2.0.3
3274                                        ; VESA VBE3 VIDE BIOS 32 BIT PMI SEGMENTS (16 bit segments)
3275                                                    ; 30h ; VBE3CS
3276                                    _vbe3_CS:  ; vesa vbe3 bios uses this as code seg (same addr with _vbe3_DS)
3277                                        ; limit = 65536, base addr = 0, P/DPL/1/Type/C/R/A = 9Ah, 16 bit
3278                                        ;dw 0FFFFh, 0, 9A00h, 0 ; Note: base addr will be initialized
3279                                        ; 04/12/2023 - TRDOS 386 v2.0.7      ; (+!*!+)
3280 000065A0 FFFF0000069A0000             dw 0FFFFh, 0, 9A06h, 0; VBE3BIOSCODE_ADDR = 60000h
3281                                                    ; 38h ; VBE3BDS
3282                                    _vbe3_BDS: ; vesa vbe3 bios uses this as equivalent of rombios data segment
3283                                        ; limit = 1536, base addr = 0, P/DPL/1/Type/E/W/A = 92h, 16 bit
3284 000065A8 FF05000000920000             dw 05FFh, 0, 9200h, 0 ; Note: base addr will be initialized
3285                                                    ; 40h ; VBE3A000
3286                                    _A0000Sel: ; VGA default video memory address
3287                                        ; limit = 65536, base addr = 0A0000h, 16 bit
3288 000065B0 FFFF00000A920000             dw 0FFFFh, 0, 920Ah, 0
3289                                                    ; 48h ; VBE3B000
3290                                    _B0000Sel: ; MDA (monochrome) video memory address
3291                                        ; limit = 65536, base addr = 0B0000h, 16 bit
3292 000065B8 FFFF00000B920000             dw 0FFFFh, 0, 920Bh, 0
3293                                                    ; 50h ; VBE3B800
3294                                    _B8000Sel: ; CGA video memory address
3295                                        ; limit = 32768, base addr = 0B8000h, 16 bit
3296 000065C0 FF7F00800B920000             dw 07FFFh, 8000h, 920Bh, 0
3297                                                    ; 58h ; VBE3DS
3298                                    _vbe3_DS: ; vesa vbe3 bios uses this as data seg (CodeSegSel in PMInfoBlock)
3299                                        ; limit = 65536, base addr = 0, P/DPL/1/Type/E/W/A = 92h, 16 bit
3300                                        ;dw 0FFFFh, 0, 9200h, 0 ; Note: base addr will be initialized
3301                                        ; 04/12/2023 - TRDOS 386 v2.0.7      ; (+!*!+)
3302 000065C8 FFFF000006920000             dw 0FFFFh, 0, 9206h, 0; VBE3BIOSCODE_ADDR = 60000h
3303                                                    ; 60h ; VBE3SS
3304                                    _vbe3_SS: ; kernel's stack segment but 16 bit version (same stack addr)
3305                                        ; limit = 1024, base addr = 0, P/DPL/1/Type/E/W/A = 92h, 16 bit
3306 000065D0 FF03000000920000             dw 03FFh, 0, 9200h, 0 ; Note: base addr will be initialized
3307                                                    ; 68h ; VBE3ES
3308                                    _vbe3_ES: ; extra 16 bit segment points to buffers in kernel's mem space
3309                                        ; limit = 2048, base addr = 0, P/DPL/1/Type/E/W/A = 92h, 16 bit
3310 000065D8 FF07000000920000             dw 07FFh, 0, 9200h,0 ; Note: base addr will be initialized
3311                                                    ; 70h ; KODE16
3312                                    _16bit_CS: ; 16 bit code segment points to kernel's far return addr
3313                                        ; limit = 16M, base addr = 0, P/DPL/1/Type/E/W/A = 92h, 16 bit
3314 000065E0 FFFF0000009AFF00             dw 0FFFFh, 0, 9A00h, 00FFh ; Note: base addr will be initialized
3315
3316                                    gdt_end:
3317                                        ;; 9Eh = 1001 1110b (GDT byte 5) P=1/DPL=00/1/TYPE=1110,
3318                                                                        ;; Type= 1 (code)/C=0/R=1/A=0
3319                                                ; P= Present, DPL=0=ring 0,  1= user (0= system)
3320                                                ; 1= Code C= Conforming, R= Readable, A= Accessed
3321
3322                                    ;; 9Ah = 1001 1010b (GDT byte 5) P=1/DPL=00/1/TYPE=1010,
3323                                                                        ;; Type= 1 (code)/C=0/R=1/A=0
3324                                            ; P= Present, DPL=0=ring 0,  1= user (0= system)
3325                                            ; 1= Code C= non-Conforming, R= Readable, A= Accessed
3326
3327                                    ;; 92h = 1001 0010b (GDT byte 5) P=1/DPL=00/1/TYPE=1010,
3328                                                                        ;; Type= 0 (data)/E=0/W=1/A=0
3329                                            ; P= Present, DPL=0=ring 0,  1= user (0= system)
3330                                            ; 0= Data E= Expansion direction (1= down, 0= up)
3331                                            ; W= Writeable, A= Accessed
3332
3333                                    ;; FEh = 1111 1110b (GDT byte 5) P=1/DPL=11/1/TYPE=1110,
3334                                                                        ;; Type= 1 (code)/C=1/R=1/A=0
3335                                            ; P= Present, DPL=3=ring 3,  1= user (0= system)
3336                                            ; 1= Code C= Conforming, R= Readable, A= Accessed
3337
3338                                    ;; FAh = 1111 1010b (GDT byte 5) P=1/DPL=11/1/TYPE=1010,
3339                                                                        ;; Type= 1 (code)/C=0/R=1/A=0
3340                                            ; P= Present, DPL=3=ring 3,  1= user (0= system)
3341                                            ; 1= Code C= non-Conforming, R= Readable, A= Accessed
3342
3343                                    ;; F2h = 1111 0010b (GDT byte 5) P=1/DPL=11/1/TYPE=0010,
3344                                                                        ;; Type= 0 (data)/E=0/W=1/A=0
3345                                            ; P= Present, DPL=3=ring 3,  1= user (0= system)
3346                                            ; 0= Data E= Expansion direction (1= down, 0= up)
3347
3348                                    ;; CFh = 1100 1111b (GDT byte 6) G=1/B=1/0/AVL=0, Limit=1111b (3)
3349
3350                                            ;; Limit = FFFFFh (=> FFFFFh+1= 100000h) // bits 0-15, 48-51 //
3351                                            ;       = 100000h * 1000h (G=1) = 4GB
3352                                            ;; Limit = FFBFFh (=> FFBFFh+1= FFC00h) // bits 0-15, 48-51 //
3353                                            ;       = FFC00h * 1000h (G=1) = 4GB - 4MB
3354                                            ; G= Granularity (1= 4KB), B= Big (32 bit),
3355                                            ; AVL= Available to programmers
```

```
                                    gdtd:
000065E8 7700                               dw gdt_end - gdt - 1    ; Limit (size)
000065EA [70650000]                         dd gdt                  ; Address of the GDT
                                    
                                        ; 20/08/2014
                                    idtd:
000065EE 7F02                               dw idt_end - idt - 1    ; Limit (size)
000065F0 [98730100]                         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) - v2.0.5 - diskdata.s
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Last Update: 06/08/2022 (Previous: 24/01/2016)
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Beginning: 24/01/2016
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Assembler: NASM version 2.15 (trdos386.s)
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Turkish Rational DOS
                                <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
                                <1> ;
                                <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
                                <1> ; 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> ;---------------------------------------
                                <1>
000065F4 [57660000]             <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:
```

```
112 000065F8 01                 <1>             DB      01              ; DRIVE TYPE, MEDIA TABLE
113                             <1>                 ;DW       MD_TBL1
114 000065F9 [16660000]         <1>             dd      MD_TBL1
115 000065FD 82                 <1>             DB      02+BIT7ON
116                             <1>             ;DW     MD_TBL2
117 000065FE [23660000]         <1>                 dd      MD_TBL2
118 00006602 02                 <1> DR_DEFAULT: DB      02
119                             <1>                 ;DW       MD_TBL3
120 00006603 [30660000]         <1>             dd       MD_TBL3
121 00006607 03                 <1>             DB      03
122                             <1>                 ;DW       MD_TBL4
123 00006608 [3D660000]         <1>             dd       MD_TBL4
124 0000660C 84                 <1>             DB      04+BIT7ON
125                             <1>                 ;DW       MD_TBL5
126 0000660D [4A660000]         <1>             dd       MD_TBL5
127 00006611 04                 <1>             DB      04
128                             <1>                 ;DW       MD_TBL6
129 00006612 [57660000]         <1>             dd       MD_TBL6
130                             <1> DR_TYPE_E       equ $                   ; END OF TABLE
131                             <1> ;DR_CNT         EQU     (DR_TYPE_E-DR_TYPE)/3
132                             <1> DR_CNT          equ     (DR_TYPE_E-DR_TYPE)/5
133                             <1> ;---------------------------------------------------------
134                             <1> ;   MEDIA/DRIVE PARAMETER TABLES               :
135                             <1> ;---------------------------------------------------------
136                             <1> ;---------------------------------------------------------
137                             <1> ;   360 KB MEDIA IN 360 KB DRIVE                :
138                             <1> ;---------------------------------------------------------
139                             <1> MD_TBL1:
140 00006616 DF                 <1>     DB      11011111B       ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
141 00006617 02                 <1>     DB      2               ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
142 00006618 25                 <1>     DB      MOTOR_WAIT      ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
143 00006619 02                 <1>     DB      2               ; 512 BYTES/SECTOR
144 0000661A 09                 <1>     DB      09              ; EOT (LAST SECTOR ON TRACK)
145 0000661B 2A                 <1>     DB      02AH            ; GAP LENGTH
146 0000661C FF                 <1>     DB      0FFH            ; DTL
147 0000661D 50                 <1>     DB      050H            ; GAP LENGTH FOR FORMAT
148 0000661E F6                 <1>     DB      0F6H            ; FILL BYTE FOR FORMAT
149 0000661F 0F                 <1>     DB      15              ; HEAD SETTLE TIME (MILLISECONDS)
150 00006620 08                 <1>     DB      8               ; MOTOR START TIME (1/8 SECONDS)
151 00006621 27                 <1>     DB      39              ; MAX. TRACK NUMBER
152 00006622 80                 <1>     DB      RATE_250        ; DATA TRANSFER RATE
153                             <1> ;---------------------------------------------------------
154                             <1> ;   360 KB MEDIA IN 1.2 MB DRIVE                :
155                             <1> ;---------------------------------------------------------
156                             <1> MD_TBL2:
157 00006623 DF                 <1>     DB      11011111B       ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
158 00006624 02                 <1>     DB      2               ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
159 00006625 25                 <1>     DB      MOTOR_WAIT      ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
160 00006626 02                 <1>     DB      2               ; 512 BYTES/SECTOR
161 00006627 09                 <1>     DB      09              ; EOT (LAST SECTOR ON TRACK)
162 00006628 2A                 <1>     DB      02AH            ; GAP LENGTH
163 00006629 FF                 <1>     DB      0FFH            ; DTL
164 0000662A 50                 <1>     DB      050H            ; GAP LENGTH FOR FORMAT
165 0000662B F6                 <1>     DB      0F6H            ; FILL BYTE FOR FORMAT
166 0000662C 0F                 <1>     DB      15              ; HEAD SETTLE TIME (MILLISECONDS)
167 0000662D 08                 <1>     DB      8               ; MOTOR START TIME (1/8 SECONDS)
168 0000662E 27                 <1>     DB      39              ; MAX. TRACK NUMBER
169 0000662F 40                 <1>     DB      RATE_300        ; DATA TRANSFER RATE
170                             <1> ;---------------------------------------------------------
171                             <1> ;   1.2 MB MEDIA IN 1.2 MB DRIVE                :
172                             <1> ;---------------------------------------------------------
173                             <1> MD_TBL3:
174 00006630 DF                 <1>     DB      11011111B       ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
175 00006631 02                 <1>     DB      2               ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
176 00006632 25                 <1>     DB      MOTOR_WAIT      ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
177 00006633 02                 <1>     DB      2               ; 512 BYTES/SECTOR
178 00006634 0F                 <1>     DB      15              ; EOT (LAST SECTOR ON TRACK)
179 00006635 1B                 <1>     DB      01BH            ; GAP LENGTH
180 00006636 FF                 <1>     DB      0FFH            ; DTL
181 00006637 54                 <1>     DB      054H            ; GAP LENGTH FOR FORMAT
182 00006638 F6                 <1>     DB      0F6H            ; FILL BYTE FOR FORMAT
183 00006639 0F                 <1>     DB      15              ; HEAD SETTLE TIME (MILLISECONDS)
184 0000663A 08                 <1>     DB      8               ; MOTOR START TIME (1/8 SECONDS)
185 0000663B 4F                 <1>     DB      79              ; MAX. TRACK NUMBER
186 0000663C 00                 <1>     DB      RATE_500        ; DATA TRANSFER RATE
187                             <1> ;---------------------------------------------------------
188                             <1> ;   720 KB MEDIA IN 720 KB DRIVE                :
189                             <1> ;---------------------------------------------------------
190                             <1> MD_TBL4:
191 0000663D DF                 <1>     DB      11011111B       ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
192 0000663E 02                 <1>     DB      2               ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
193 0000663F 25                 <1>     DB      MOTOR_WAIT      ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
194 00006640 02                 <1>     DB      2               ; 512 BYTES/SECTOR
195 00006641 09                 <1>     DB      09              ; EOT (LAST SECTOR ON TRACK)
196 00006642 2A                 <1>     DB      02AH            ; GAP LENGTH
197 00006643 FF                 <1>     DB      0FFH            ; DTL
198 00006644 50                 <1>     DB      050H            ; GAP LENGTH FOR FORMAT
199 00006645 F6                 <1>     DB      0F6H            ; FILL BYTE FOR FORMAT
200 00006646 0F                 <1>     DB      15              ; HEAD SETTLE TIME (MILLISECONDS)
201 00006647 08                 <1>     DB      8               ; MOTOR START TIME (1/8 SECONDS)
202 00006648 4F                 <1>     DB      79              ; MAX. TRACK NUMBER
203 00006649 80                 <1>     DB      RATE_250        ; DATA TRANSFER RATE
204                             <1> ;---------------------------------------------------------
205                             <1> ;   720 KB MEDIA IN 1.44 MB DRIVE               :
206                             <1> ;---------------------------------------------------------
207                             <1> MD_TBL5:
208 0000664A DF                 <1>     DB      11011111B       ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
209 0000664B 02                 <1>     DB      2               ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
210 0000664C 25                 <1>     DB      MOTOR_WAIT      ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
211 0000664D 02                 <1>     DB      2               ; 512 BYTES/SECTOR
212 0000664E 09                 <1>     DB      09              ; EOT (LAST SECTOR ON TRACK)
213 0000664F 2A                 <1>     DB      02AH            ; GAP LENGTH
214 00006650 FF                 <1>     DB      0FFH            ; DTL
215 00006651 50                 <1>     DB      050H            ; GAP LENGTH FOR FORMAT
216 00006652 F6                 <1>     DB      0F6H            ; FILL BYTE FOR FORMAT
217 00006653 0F                 <1>     DB      15              ; HEAD SETTLE TIME (MILLISECONDS)
218 00006654 08                 <1>     DB      8               ; MOTOR START TIME (1/8 SECONDS)
219 00006655 4F                 <1>     DB      79              ; MAX. TRACK NUMBER
220 00006656 80                 <1>     DB      RATE_250        ; DATA TRANSFER RATE
221                             <1> ;---------------------------------------------------------
222                             <1> ;   1.44 MB MEDIA IN 1.44 MB DRIVE                 :
223                             <1> ;---------------------------------------------------------
224                             <1> MD_TBL6:
225 00006657 AF                 <1>     DB      10101111B       ; SRT=A, HD UNLOAD=0F - 1ST SPECIFY BYTE
226 00006658 02                 <1>     DB      2               ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
227 00006659 25                 <1>     DB      MOTOR_WAIT      ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
228 0000665A 02                 <1>     DB      2               ; 512 BYTES/SECTOR
229 0000665B 12                 <1>     DB      18              ; EOT (LAST SECTOR ON TRACK)
230 0000665C 1B                 <1>     DB      01BH            ; GAP LENGTH
231 0000665D FF                 <1>     DB      0FFH            ; DTL
232 0000665E 6C                 <1>     DB      06CH            ; GAP LENGTH FOR FORMAT
233 0000665F F6                 <1>     DB      0F6H            ; FILL BYTE FOR FORMAT
234 00006660 0F                 <1>     DB      15              ; HEAD SETTLE TIME (MILLISECONDS)
235 00006661 08                 <1>     DB      8               ; MOTOR START TIME (1/8 SECONDS)
```

```
00006662 4F                     <1>     DB      79              ; MAX. TRACK NUMBER
00006663 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:
00006664 E0                     <1>     db      NO_ERR
00006665 024001BB               <1>     db      BAD_ADDR_MARK,BAD_SEEK,BAD_CMD,UNDEF_ERR
00006669 04BB100A               <1>     db      RECORD_NOT_FND,UNDEF_ERR,BAD_ECC,BAD_SECTOR
                                <1>
                                <1> ; 06/08/2022
                                <1> ; 17/12/2014 (mov ax, [cfd])
                                <1> ; 11/12/2014
                                <1> ;cfd:               db      0               ; current floppy drive (for GET_PARM)
                                <1> ; 17/12/2014                                ; instead of 'DISK_POINTER'
                                <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)
                                <1> ; 06/08/2022 - TRDOS 386 Kernel v2.0.5
                                <1> ; 11/07/2022 - Retro UNIX 386 v1.1 (Kernel v0.2.1.5)
0000666D FF                     <1> pfd:        db      0FFh
                                <1>
                                <1> align 2
                                <1>
0000666E F001                   <1> HF_PORT:    dw      1F0h  ; Default = 1F0h
                                <1>                             ; (170h)
00006670 F603                   <1> HF_REG_PORT:        dw      3F6h  ; HF_PORT + 206h
                                <1>
                                <1> ; 05/01/2015
00006672 00                     <1> hf_m_s:         db      0     ; (0 = Master, 1 = Slave)
                                <1>
                                <1> ; ****************************************************************************
                                
00006673 90                         Align 2
                                
                                    ; 04/11/2014 (Retro UNIX 386 v1)
00006674 0000                       mem_1m_1k:   dw 0  ; Number of contiguous KB between
                                                       ; 1 and 16 MB, max. 3C00h = 15 MB.
00006676 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)
00006678 00                         boot_drv:    db 0 ; boot drive number (physical)
                                    ; 24/11/2014
00006679 00                         drv:     db 0
0000667A 00                         last_drv:    db 0 ; last hdd
0000667B 00                         hdc:         db 0  ; number of hard disk drives
                                                     ; (present/detected)
                                
                                    ; 24/11/2014 (Retro UNIX 386 v1)
                                    ; Physical drive type & flags
0000667C 00                         fd0_type:    db 0  ; floppy drive type
0000667D 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)
0000667E 00                         hd0_type:    db 0  ; EDD status for hd0 (bit 7 = present flag)
0000667F 00                         hd1_type:    db 0  ; EDD status for hd1 (bit 7 = present flag)
00006680 00                         hd2_type:    db 0  ; EDD status for hd2 (bit 7 = present flag)
00006681 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'!)
                                
                                    ; 08/08/2022
                                    ; (drv.cylinders, drv.spt, drv.spt will not be used now on)
                                    ; ('diskio.inc')
                                    ; ((spt and heads and cylinder counts will be taken from DPT))
                                
                                    ;; 11/03/2015 - 10/07/2015
                                    ;drv.cylinders: dw 0,0,0,0,0,0,0,0
                                    ;drv.heads:     dw 0,0,0,0,0,0,0,0
                                    ;drv.spt:       dw 0,0,0,0,0,0,0,0
                                    ; 12/07/2022 - 11/03/2015
00006682 00000000000000000000-      drv.size:      dd 0,0,0,0,0,0,0,0
0000668B 00000000000000000000-
00006694 00000000000000000000-
0000669D 00
0000669E 00000000000000         drv.status:    db 0,0,0,0,0,0,0,0
000066A5 00000000000000         drv.error:     db 0,0,0,0,0,0,0,0
                                ;
```

```
                            
                            Align 2
                            
                            ;;; 11/03/2015
                            %include 'kybdata.s'      ; KEYBOARD (BIOS) DATA
                          <1> ; ****************************************************************************
                          <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.5 - kybdata.s
                          <1> ; ----------------------------------------------------------------------------
                          <1> ; Last Update: 24/07/2022 (Previous: 17/01/2016)
                          <1> ; ----------------------------------------------------------------------------
                          <1> ; Beginning: 17/01/2016
                          <1> ; ----------------------------------------------------------------------------
                          <1> ; Assembler: NASM version 2.15 (trdos386.s)
                          <1> ; ----------------------------------------------------------------------------
                          <1> ; Turkish Rational DOS
                          <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
                          <1> ;
                          <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
                          <1> ; 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> ; 24/07/2022 - TRDOS 386 Kernel v2.0.5
                          <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
000066AC 524F50514B       <1> K30:db      82,79,80,81,75
000066B1 4C4D474849       <1>     db      76,77,71,72,73          ; 10 NUMBER ON KEYPAD
                          <1> ;-----      SUPER-SHIFT-TABLE
000066B6 101112131415     <1>     db      16,17,18,19,20,21       ; A-Z TYPEWRITER CHARS
000066BC 161718191E1F     <1>     db      22,23,24,25,30,31
000066C2 202122232425     <1>     db      32,33,34,35,36,37
000066C8 262C2D2E2F30     <1>     db      38,44,45,46,47,48
000066CE 3132             <1>     db      49,50
                          <1>
                          <1> ;-----      TABLE OF SHIFT KEYS AND MASK VALUES
                          <1> ;-----      KEY_TABLE
000066D0 52               <1> _K6:    db      INS_KEY                 ; INSERT KEY
000066D1 3A4546381D       <1>     db      CAPS_KEY,NUM_KEY,SCROLL_KEY,ALT_KEY,CTL_KEY
000066D6 2A36             <1>         db      LEFT_KEY,RIGHT_KEY
                          <1> _K6L    equ     $-_K6
                          <1>
                          <1> ;-----      MASK_TABLE
000066D8 80               <1> _K7:    db      INS_SHIFT               ; INSERT MODE SHIFT
000066D9 4020100804       <1>     db      CAPS_SHIFT,NUM_SHIFT,SCROLL_SHIFT,ALT_SHIFT,CTL_SHIFT
000066DE 0201             <1>     db      LEFT_SHIFT,RIGHT_SHIFT
                          <1>
                          <1> ;-----      TABLES FOR CTRL CASE          ;---- CHARACTERS ------
000066E0 1BFF00FFFFFF     <1> _K8:db      27,-1,0,-1,-1,-1        ; Esc, 1, 2, 3, 4, 5
000066E6 1EFFFFFFFF1F     <1>     db      30,-1,-1,-1,-1,31       ; 6, 7, 8, 9, 0, -
                          <1>     ;db     -1,127,-1,17,23,5       ; =, Bksp, Tab, Q, W, E
000066EC FF7F94111705     <1>     db      -1,127,148,17,23,5 ; 24/07/2022
000066F2 12141915090F     <1>     db      18,20,25,21,9,15        ; R, T, Y, U, I, O
000066F8 101B1D0AFF01     <1>     db      16,27,29,10,-1,1        ; P, [, ], Enter, Ctrl, A
000066FE 13040607080A     <1>     db      19,4,6,7,8,10           ; S, D, F, G, H, J
00006704 0B0CFFFFFFFF     <1>     db      11,12,-1,-1,-1,-1       ; K, L, :, ', `, LShift
0000670A 1C1A18031602     <1>     db      28,26,24,3,22,2                 ; Bkslash, Z, X, C, V, B
00006710 0E0DFFFFFFFF     <1>     db      14,13,-1,-1,-1,-1       ; N, M, ,, ., /, RShift
00006716 96FF20FF         <1>     db      150,-1,' ',-1           ; *, ALT, Spc, CL
                          <1>     ;                               ;----- FUNCTIONS ------
0000671A 5E5F60616263     <1>     db      94,95,96,97,98,99       ; F1 - F6
00006720 64656667FFFF     <1>     db      100,101,102,103,-1,-1 ; F7 - F10, NL, SL
00006726 778D848E738F     <1>     db      119,141,132,142,115,143         ; Home, Up, PgUp, -, Left, Pad5
0000672C 749075917692     <1>     db      116,144,117,145,118,146 ; Right, +, End, Down, PgDn, Ins
00006732 93FFFFF898A      <1>     db      147,-1,-1,-1,137,138    ; Del, SysReq, Undef, WT, F11, F12
                          <1>
                          <1> ;-----      TABLES FOR LOWER CASE ----------
00006738 1B3132333435363738- <1> K10:db     27,'1234567890-=',8,9
00006741 39302D3D0809     <1>
00006747 7177657274975696F- <1>     db      'qwertyuiop[]',13,-1,'asdfghjkl;',39
00006750 705B5D0DFF61736466- <1>
00006759 67686A6B6C3B27   <1>
00006760 60FF5C7A786376626E- <1>    db      96,-1,92,'zxcvbnm,./',-1,'*',-1,' ',-1
00006769 6D2C2E2FFF2AFF20FF  <1>
                          <1> ;-----      LC TABLE SCAN
00006772 3B3C3D3E3F       <1>     db      59,60,61,62,63          ; BASE STATE OF F1 - F10
00006777 4041424344       <1>     db      64,65,66,67,68
0000677C FFFF             <1>     db      -1,-1                   ; NL, SL
                          <1>
                          <1> ;-----      KEYPAD TABLE
0000677E 474849FF4BFF     <1> K15:db      71,72,73,-1,75,-1       ; BASE STATE OF KEYPAD KEYS
00006784 4DFF4F50515253   <1>     db      77,-1,79,80,81,82,83
0000678B FFFF5C8586       <1>     db      -1,-1,92,133,134        ; SysRq, Undef, WT, F11, F12
                          <1>
                          <1> ;-----      TABLES FOR UPPER CASE ----------
00006790 1B21402324255E262A- <1> K11:db     27,'!@#$%',94,'&*()_+',8,0
00006799 28295F2B0800     <1>
0000679F 51574552545955494F- <1>    db      'QWERTYUIOP{}',13,-1,'ASDFGHJKL:"'
000067A8 507B7D0DFF41534446- <1>
000067B1 47484A4B4C3A22   <1>
000067B8 7EFF7C5A584356424E- <1>    db      126,-1,'|ZXCVBNM<>?',-1,'*',-1,' ',-1
000067C1 4D3C3E3FFF2AFF20FF  <1>
                          <1> ;-----      UC TABLE SCAN
000067CA 5455565758       <1> K12:db      84,85,86,87,88          ; SHIFTED STATE OF F1 - F10
000067CF 595A5B5C5D       <1>     db      89,90,91,92,93
000067D4 FFFF             <1>     db      -1,-1                   ; NL, SL
                          <1>
                          <1> ;-----      NUM STATE TABLE
000067D6 3738392D3435362B31- <1> K14:db     '789-456+1230.'                 ; NUMLOCK STATE OF KEYPAD KEYS
000067DF 3233302E         <1>
                          <1>     ;
000067E3 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> ;----------------------------------------
000067E8 00                 <1> BIOS_BREAK  db     0             ; BIT 7=1 IF BREAK KEY HAS BEEN PRESSED
                            <1>
                            <1> ;----------------------------------------
                            <1> ;   KEYBOARD DATA AREAS          ;
                            <1> ;----------------------------------------
                            <1>
000067E9 00                 <1> KB_FLAG            db     0             ; KEYBOARD SHIFT STATE AND STATUS FLAGS
000067EA 00                 <1> KB_FLAG_1   db     0             ; SECOND BYTE OF KEYBOARD STATUS
000067EB 00                 <1> KB_FLAG_2   db     0             ; KEYBOARD LED FLAGS
000067EC 00                 <1> KB_FLAG_3   db     0             ; KEYBOARD MODE STATE AND TYPE FLAGS
000067ED 00                 <1> ALT_INPUT   db     0             ; STORAGE FOR ALTERNATE KEY PAD ENTRY
000067EE [FE670000]         <1> BUFFER_START       dd     KB_BUFFER      ; OFFSET OF KEYBOARD BUFFER START
000067F2 [1E680000]         <1> BUFFER_END  dd     KB_BUFFER + 32 ; OFFSET OF END OF BUFFER
000067F6 [FE670000]         <1> BUFFER_HEAD dd     KB_BUFFER      ; POINTER TO HEAD OF KEYBOARD BUFFER
000067FA [FE670000]         <1> BUFFER_TAIL dd     KB_BUFFER      ; POINTER TO TAIL OF KEYBOARD BUFFER
                            <1> ; ------    HEAD = TAIL INDICATES THAT THE BUFFER IS EMPTY
000067FE 0000<rep 10h>      <1> KB_BUFFER   times  16 dw 0        ; ROOM FOR 16 SCAN CODE ENTRIES
                            <1>
                            <1> ; /// End Of KEYBOARD DATA ///
                                %include 'vidata.s'       ; VIDEO (BIOS) DATA
                            <1> ; ********************************************************************************
                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.10 - vidata.s
                            <1> ; --------------------------------------------------------------------------------
                            <1> ; Last Update: 21/12/2025 (Previous: 17/10/2023 - Kernel v2.0.7)
                            <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> ;----------------------------------------
0000681E 03                 <1> CRT_MODE:   db     3       ; CURRENT DISPLAY MODE (TYPE)
0000681F 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
00006820 50                 <1> CRT_COLS:   db     80      ; Number of columns
                            <1>
                            <1> ; 01/07/2016
00006821 00                 <1> CRT_PALETTE:       db     0       ; Current palette setting
                            <1>
                            <1> ; 03/07/2016
00006822 10                 <1> CHAR_HEIGHT:       db     16      ; Default character height
00006823 60                 <1> VGA_VIDEO_CTL:     db     60h     ; ROM BIOS DATA AREA Offset 87h
00006824 F9                 <1> VGA_SWITCHES:      db     0F9h    ; Feature Bit Switches (the basic screen)
00006825 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
00006826 19                 <1> VGA_ROWS:   db     25
                            <1>
                            <1> ; 16/01/2016
                            <1> chr_attrib:  ; Character color/attributes for video pages (0 to 7)
00006827 0707070707070707   <1>     db     07h, 07h, 07h, 07h, 07h, 07h, 07h, 07h
                            <1> ; 30/01/2016
                            <1> vmode:
0000682F 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
00006837 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>
00006839 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)
0000683A 0302010007040506   <1>     db     03h, 02h, 01h, 00h, 07h, 04h, 05h, 06h
                            <1> vga_g_modes: ; 31/07/2016
00006842 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
```

```
103 0000684A [AA680000]          <1>     dd      vga_mode_03h
104 0000684E [AA680000]          <1>     dd      vga_mode_03h ; mode 02h -> mode 03h
105 00006852 [EA680000]          <1>     dd      vga_mode_01h
106 00006856 [EA680000]          <1>     dd      vga_mode_01h ; mode 00h -> mode 01h
107                              <1>     ;dd     vga_mode_07h
108 0000685A [AA680000]          <1>     dd      vga_mode_03h ; mode 07h -> mode 03h
109 0000685E [2A690000]          <1>     dd      vga_mode_04h
110 00006862 [2A690000]          <1>     dd      vga_mode_04h ; mode 05h -> mode 04h
111 00006866 [6A690000]          <1>     dd      vga_mode_06h
112                              <1> vga_g_mode_tbl_ptr: ; 21/12/2025
113 0000686A [AA690000]          <1>     dd      vga_mode_13h
114 0000686E [EA690000]          <1>     dd      vga_mode_F0h
115 00006872 [2A6A0000]          <1>     dd      vga_mode_12h
116 00006876 [6A6A0000]          <1>     dd      vga_mode_6Ah
117 0000687A [AA6A0000]          <1>     dd      vga_mode_0Dh
118 0000687E [EA6A0000]          <1>     dd      vga_mode_0Eh
119 00006882 [2A6B0000]          <1>     dd      vga_mode_10h
120 00006886 [6A6B0000]          <1>     dd      vga_mode_11h
121                              <1>
122                              <1> vga_memmodel:
123                              <1>     ; 25/07/2016
124                              <1>     ; 07/07/2016
125                              <1>     CTEXT  equ 0
126                              <1>     ;MTEXT  equ 1
127                              <1>     MTEXT  equ 0 ; mode 07h -> mode 03h
128                              <1>     CGA    equ 2
129                              <1>     LINEAR8 equ 5
130                              <1>     PLANAR4 equ 4
131                              <1>     PLANAR1 equ 3
132 0000688A 0000000000020202   <1>     db      CTEXT, CTEXT, CTEXT, CTEXT, MTEXT, CGA, CGA, CGA
133                              <1> vga_g_memmodel: ; 31/07/2016
134 00006892 0504040404040403   <1>     db      LINEAR8, PLANAR4, PLANAR4, PLANAR4, PLANAR4, PLANAR4, PLANAR4, PLANAR1
135                              <1> ;vga_pixbits:
136                              <1> ;   ; 25/07/2016
137                              <1> ;   ; 08/07/2016
138                              <1> ;   db      4, 4, 4, 4, 4, 4, 2, 2, 1, 8, 4, 4, 4, 4, 4, 4, 1
139                              <1> vga_dac_s:
140 0000689A 020202020001010103- <1>     db      2, 2, 2, 2, 0, 1, 1, 1, 3, 3, 2, 2, 1, 1, 2, 2
141 000068A3 03020201010202     <1>
142                              <1>                                             ; (vgatables.h, VGAMODES, dac)
143                              <1>                                             ; 17/11/2020
144                              <1> vga_params:
145                              <1>     ; 23/11/2020
146                              <1>     ; 16/11/2020
147                              <1>     ; 09/11/2020, 10/11/2020, 11/11/2020 (TRDOS 386 v2.0.3)
148                              <1>     ; 25/07/2016
149                              <1>     ; 19/07/2016
150                              <1>     ; 03/07/2016
151                              <1>     ; derived from 'Plex86/Bochs VGABios' source code
152                              <1>     ; vgabios-0.7a (2011)
153                              <1>     ; by the LGPL VGABios Developers Team (2001-2008)
154                              <1>     ; 'vgatables.h'
155                              <1>     ; Oracle VirtualBox 5.0.24 VGABios Source Code
156                              <1>     ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
157                              <1>
158                              <1>     ; 09/11/2020
159                              <1>     ; Block Structure of Video Parameter Table
160                              <1>     ;
161                              <1>     ; Offset       # Bytes        Contents
162                              <1>     ;
163                              <1>     ;  0           1              # columns
164                              <1>     ;  1           1              # rows - 1
165                              <1>     ;  2           1              Pixels/character
166                              <1>     ;  3-4         2              Page length
167                              <1>     ;  5-8         4              Sequencer Registers
168                              <1>     ;  9           1              Miscellaneous Register
169                              <1>     ;  10-34       25             CRTC Registers
170                              <1>     ;  35-54       20             Attribute Registers
171                              <1>     ;  55-63       9              Graphics Controller Registers
172                              <1>     ;
173                              <1>     ; Ref: Programmer's Guide to EGA, VGA, and Super VGA cards
174                              <1>     ; (Richard F. Ferraro, 1994)
175                              <1>     ;
176                              <1> vga_mode_03h:  ; mode 03h, 80*25 text, CGA colors
177                              <1>     ; 11/11/2020
178 000068AA 5018100010         <1>     db      80, 24, 16, 00h, 10h ; tw, th-1, ch, slength (5)
179 000068AF 00030002           <1>     db      00h, 03h, 00h, 02h ; sequ regs (4)
180 000068B3 67                 <1>     db      67h     ; misc reg (1)
181 000068B4 5F4F50825581BF1F   <1>     db      5Fh, 4Fh, 50h, 82h, 55h, 81h, 0BFh, 1Fh
182                              <1>     ; 09/11/2020
183                              <1>     ;db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
184 000068BC 004F               <1>     db      00h, 4Fh
185                              <1> vga_p_cm_pos equ $ - vga_mode_03h
186 000068BE 0D0E00000000       <1>     db      0Dh, 0Eh, 00h, 00h, 00h, 00h
187 000068C4 9C8E8F281F96B9A3   <1>     db      9Ch, 8Eh, 8Fh, 28h, 1Fh, 96h, 0B9h, 0A3h
188 000068CC FF                 <1>     db      0FFh    ; crtc_regs (25)
189 000068CD 0001020304051407   <1>     db      00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
190 000068D5 38393A3B3C3D3E3F   <1>     db      38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
191                              <1>     ; 17/10/2023
192 000068DD 0C000F08           <1>     db      0Ch, 00h, 0Fh, 08h  ; actl regs (20)
193                              <1>     ;db     0Ch, 00h, 0Fh, 00h  ; 19/11/2020
194 000068E1 0000000000100E0FFF  <1>     db      00h, 00h, 00h, 00h, 00h, 10h, 0Eh, 0Fh, 0FFh ; grdc regs (9)
195                              <1>     ; 09/11/2020
196                              <1>     ;db     00h, 00h, 00h, 00h, 00h, 10h, 0Eh, 00h, 0FFh ; grdc regs (9)
197                              <1> vga_mode_01h:      ; mode 01h, 40*25 text, CGA colors
198 000068EA 2818100008         <1>     db      40, 24, 16, 00h, 08h ; tw, th-1, ch, slength
199 000068EF 08030002           <1>     db      08h, 03h, 00h, 02h  ; sequ regs
200 000068F3 67                 <1>     db      67h     ; misc reg
201 000068F4 2D2728902BA0BF1F   <1>     db      2Dh, 27h, 28h, 90h, 2Bh, 0A0h, 0BFh, 1Fh
202 000068FC 004F0D0E00000000   <1>     db      00h, 4Fh, 0Dh, 0Eh, 00h, 00h, 00h, 00h
203 00006904 9C8E8F141F96B9A3   <1>     db      9Ch, 8Eh, 8Fh, 14h, 1Fh, 96h, 0B9h, 0A3h
204 0000690C FF                 <1>     db      0FFh    ; crtc_regs
205 0000690D 0001020304051407   <1>     db      00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
206 00006915 38393A3B3C3D3E3F   <1>     db      38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
207                              <1>     ;db     0Ch, 00h, 0Fh, 08h  ; actl regs (20)
208 0000691D 0C000F00           <1>     db      0Ch, 00h, 0Fh, 00h  ; 19/11/2020
209 00006921 0000000000100E0FFF  <1>     db      00h, 00h, 00h, 00h, 00h, 10h, 0Eh, 0Fh, 0FFh ; grdc regs
210                              <1> ;vga_mode_07h:     ; mode 07h, 80*25 text, mono color
211                              <1> ;   db      80, 24, 16, 00h, 10h  ; tw, th-1, ch, slength
212                              <1> ;   db      00h, 03h, 00h, 02h ; sequ regs
213                              <1> ;   db      66h     ; misc reg
214                              <1> ;   db      5Fh, 4Fh, 50h, 82h, 55h, 81h, 0BFh, 1Fh
215                              <1> ;   db      00h, 4Fh, 0Dh, 0Eh, 00h, 00h, 00h, 00h
216                              <1> ;   db      9Ch, 8Eh, 8Fh, 28h, 0Fh, 96h, 0B9h, 0A3h
217                              <1> ;   db      0FFh    ; crtc regs
218                              <1> ;   db      00h, 08h, 08h, 08h, 08h, 08h, 08h, 08h
219                              <1> ;   db      10h, 18h, 18h, 18h, 18h, 18h, 18h, 18h
220                              <1> ;   db      0Eh, 00h, 0Fh, 08h  ; actl regs
221                              <1> ;   db      00h, 00h, 00h, 00h, 00h, 10h, 0Ah, 0Fh, 0FFh ; grdc regs
222                              <1> vga_mode_04h:      ; 320*200 graphics, 4 colors, CGA
223                              <1>     ; 11/11/2020
224 0000692A 2818080040         <1>     db      40, 24, 8, 00h, 40h ; tw, th-1, ch, slength
225 0000692F 09030002           <1>     db      09h, 03h, 00h, 02h ; sequ regs
```

```
00006933 63                  <1>     db      63h     ; misc reg
00006934 2D2728902B80BF1F    <1>     db      2Dh, 27h, 28h, 90h, 2Bh, 80h, 0BFh, 1Fh
0000693C 00C1000000000000    <1>     db      00h, 0C1h, 00h, 00h, 00h, 00h, 00h, 00h
00006944 9C8E8F140096B9A2    <1>     db      9Ch, 8Eh, 8Fh, 14h, 00h, 96h, 0B9h, 0A2h
0000694C FF                  <1>     db      0FFh    ; crtc_regs
0000694D 0013151702040607    <1>     db      00h, 13h, 15h, 17h, 02h, 04h, 06h, 07h
00006955 1011121314151617    <1>     db      10h, 11h, 12h, 13h, 14h, 15h, 16h, 17h
0000695D 01000300            <1>     db      01h, 00h, 03h, 00h ; actl regs
00006961 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
0000696A 5018080040          <1>     db      80, 24, 8, 00h, 40h   ;       tw, th-1, ch, slength
0000696F 01010006            <1>     db      01h, 01h, 00h, 06h ; sequ regs
00006973 63                  <1>     db      63h     ; misc reg
00006974 5F4F50825480BF1F    <1>     db      5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
0000697C 00C1000000000000    <1>     db      00h, 0C1h, 00h, 00h, 00h, 00h, 00h, 00h
00006984 9C8E8F280096B9C2    <1>     db      9Ch, 8Eh, 8Fh, 28h, 00h, 96h, 0B9h, 0C2h
0000698C FF                  <1>     db      0FFh    ; crtc regs
0000698D 0017171717171717    <1>     db      00h, 17h, 17h, 17h, 17h, 17h, 17h, 17h
00006995 1717171717171717    <1>     db      17h, 17h, 17h, 17h, 17h, 17h, 17h, 17h
0000699D 01000100            <1>     db      01h, 00h, 01, 00h  ; actl regs
000069A1 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
000069AA 28180800FA          <1>     db      40, 24, 8, 00h, 0FAh ; tw, th-1, ch, slength (5)
000069AF 010F000E            <1>     db      01h, 0Fh, 00h, 0Eh ; sequ regs (4)
000069B3 63                  <1>     db      63h     ; misc reg (1)
000069B4 5F4F50825480BF1F    <1>     db      5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
000069BC 0041000000000000    <1>     db      00h, 041h, 00h, 00h, 00h, 00h, 00h, 00h
000069C4 9C8E8F284096B9A3    <1>     db      9Ch, 8Eh, 8Fh, 28h, 40h, 96h, 0B9h, 0A3h
                             <1>     ;db     9Ch, 0Eh, 8Fh, 28h, 40h, 96h, 0B9h, 0A3h ; 17/10/2023
000069CC FF                  <1>     db      0FFh    ; crtc regs (25)
000069CD 0001020304050607    <1>     db      00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
000069D5 08090A0B0C0D0E0F    <1>     db      08h, 09h, 0Ah, 0Bh, 0Ch, 0Dh, 0Eh, 0Fh
000069DD 41000F00            <1>     db      41h, 00h, 0Fh, 00h  ; actl regs (20)
                             <1>     ; 10/11/2020
                             <1>     ;db     41h, 01h, 0Fh, 13h  ; actl regs (20)
000069E1 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
000069EA 2818080000          <1>     db      40, 24, 8, 00h, 00h ; tw, th-1, ch, slength
000069EF 010F0006            <1>     db      01h, 0Fh, 00h, 06h ; sequ regs
000069F3 E3                  <1>     db      0E3h    ; misc reg
000069F4 5F4F508254800D3E    <1>     db      5Fh, 4Fh, 50h, 82h, 54h, 80h, 0Dh, 3Eh
000069FC 0041000000000000    <1>     db      00h, 41h, 00h, 00h, 00h, 00h, 00h, 00h
00006A04 EAACDF2800E706E3    <1>     db      0EAh, 0ACh, 0DFh, 28h, 00h, 0E7h, 06h, 0E3h
00006A0C FF                  <1>     db      0FFh    ; crtc regs (25)
00006A0D 0001020304050607    <1>     db      00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
00006A15 08090A0B0C0D0E0F    <1>     db      08h, 09h, 0Ah, 0Bh, 0Ch, 0Dh, 0Eh, 0Fh
00006A1D 41000F00            <1>     db      41h, 00h, 0Fh, 00h  ; actl regs
00006A21 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
00006A2A 501D1000A0          <1>     db      80, 29, 16, 00h, 0A0h ; tw, th-1, ch, slength
00006A2F 010F0006            <1>     db      01h, 0Fh, 00h, 06h ; sequ regs
00006A33 E3                  <1>     db      0E3h    ; misc reg
00006A34 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
00006A3C 0040000000000000    <1>     db      00h, 40h, 00h, 00h, 00h, 00h, 00h, 00h
00006A44 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
00006A4C FF                  <1>     db      0FFh    ; crtc regs
00006A4D 0001020304051407    <1>     db      00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
00006A55 38393A3B3C3D3E3F    <1>     db      38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
00006A5D 01000F00            <1>     db      01h, 00h, 0Fh, 00h  ; actl regs
00006A61 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
00006A6A 6424100000          <1>     db      100, 36, 16, 00h, 00h ; tw, th-1, ch, slength
00006A6F 010F0006            <1>     db      01h, 0Fh, 00h, 06h ; sequ regs
00006A73 E3                  <1>     db      0E3h    ; misc reg
00006A74 7F6363836B1B72F0    <1>     db      7Fh, 63h, 63h, 83h, 6Bh, 1Bh, 72h, 0F0h
00006A7C 0060000000000000    <1>     db      00h, 60h, 00h, 00h, 00h, 00h, 00h, 00h
00006A84 598D5732005773E3    <1>     db      59h, 8Dh, 57h, 32h, 00h, 57h, 73h, 0E3h
00006A8C FF                  <1>     db      0FFh    ; crtc regs
00006A8D 0001020304051407    <1>     db      00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
00006A95 38393A3B3C3D3E3F    <1>     db      38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
00006A9D 01000F00            <1>     db      01h, 00h, 0Fh, 00h  ; actl regs
00006AA1 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
00006AAA 2818080020          <1>     db      40, 24, 8, 00h, 20h ; tw, th-1, ch, slength
00006AAF 090F0006            <1>     db      09h, 0Fh, 00h, 06h ; sequ regs
00006AB3 63                  <1>     db      63h     ; misc reg
00006AB4 2D2728902B80BF1F    <1>     db      2Dh, 27h, 28h, 90h, 2Bh, 80h, 0BFh, 1Fh
00006ABC 00C0000000000000    <1>     db      00h, 0C0h, 00h, 00h, 00h, 00h, 00h, 00h
00006AC4 9C8E8F140096B9E3    <1>     db      9Ch, 8Eh, 8Fh, 14h, 00h, 96h, 0B9h, 0E3h
00006ACC FF                  <1>     db      0FFh    ; crtc regs
00006ACD 0001020304050607    <1>     db      00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
00006AD5 1011121314151617    <1>     db      10h, 11h, 12h, 13h, 14h, 15h, 16h, 17h
00006ADD 01000F00            <1>     db      01h, 00h, 0Fh, 00h  ; actl regs
00006AE1 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
00006AEA 5018080040          <1>     db      80, 24, 8, 00h, 40h ; tw, th-1, ch, slength
00006AEF 010F0006            <1>     db      01h, 0Fh, 00h, 06h ; sequ regs
00006AF3 63                  <1>     db      63h     ; misc reg
00006AF4 5F4F50825480BF1F    <1>     db      5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
00006AFC 00C0000000000000    <1>     db      00h, 0C0h, 00h, 00h, 00h, 00h, 00h, 00h
00006B04 9C8E8F280096B9E3    <1>     db      9Ch, 8Eh, 8Fh, 28h, 00h, 96h, 0B9h, 0E3h
00006B0C FF                  <1>     db      0FFh    ; crtc regs
00006B0D 0001020304050607    <1>     db      00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
00006B15 1011121314151617    <1>     db      10h, 11h, 12h, 13h, 14h, 15h, 16h, 17h
00006B1D 01000F00            <1>     db      01h, 00h, 0Fh, 00h  ; actl regs
00006B21 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
00006B2A 50180E0080          <1>     db      80, 24, 14, 00h, 80h ; tw, th-1, ch, slength
00006B2F 010F0006            <1>     db      01h, 0Fh, 00h, 06h ; sequ regs
00006B33 A3                  <1>     db      0A3h    ; misc reg
00006B34 5F4F50825480BF1F    <1>     db      5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
00006B3C 0040000000000000    <1>     db      00h, 40h, 00h, 00h, 00h, 00h, 00h, 00h
00006B44 83855D280F63BAE3    <1>     db      83h, 85h, 5Dh, 28h, 0Fh, 63h, 0BAh, 0E3h
00006B4C FF                  <1>     db      0FFh    ; crtc regs
00006B4D 0001020304051407    <1>     db      00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
00006B55 38393A3B3C3D3E3F    <1>     db      38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
00006B5D 01000F00            <1>     db      01h, 00h, 0Fh, 00h  ; actl regs
00006B61 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
00006B6A 501D1000A0          <1>     db      80, 29, 16, 00h, 0A0h ; tw, th-1, ch, slength
```

```
00006B6F 010F0006            <1>     db      01h, 0Fh, 00h, 06h ; sequ regs
00006B73 E3                  <1>     db      0E3h    ; misc reg
00006B74 5F4F508254800B3E    <1>     db      5Fh, 4Fh, 50h, 82h, 54h, 80h, 0Bh, 3Eh
00006B7C 0040000000000000    <1>     db      00h, 40h, 00h, 00h, 00h, 00h, 00h, 00h
00006B84 EA8CDF2800E704C3    <1>     db      0EAh, 8Ch, 0DFh, 28h, 00h, 0E7h, 04h, 0C3h ; 11/11/2020
00006B8C FF                  <1>     db      0FFh    ; crtc regs
00006B8D 003F003F003F003F    <1>     db      00h, 3Fh, 00h, 3Fh, 00h, 3Fh, 00h, 3Fh
00006B95 003F003F003F003F    <1>     db      00h, 3Fh, 00h, 3Fh, 00h, 3Fh, 00h, 3Fh
00006B9D 01000F00            <1>     db      01h, 00h, 0Fh, 00h  ; actl regs
00006BA1 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 */
00006BAA 0001800290010800    <1>     dw 100h,  640, 400,  8
00006BB2 01018002E0010800    <1>     dw 101h,  640, 480,  8
00006BBA 0301200358020800    <1>     dw 103h,  800, 600,  8
00006BC2 0501000400030800    <1>     dw 105h, 1024, 768,  8
00006BCA 0E014001C8001000    <1>     dw 10Eh,  320, 200, 16
00006BD2 0F014001C8001800    <1>     dw 10Fh,  320, 200, 24
00006BDA 11018002E0011000    <1>     dw 111h,  640, 480, 16
00006BE2 12018002E0011800    <1>     dw 112h,  640, 480, 24
00006BEA 1401200358021000    <1>     dw 114h,  800, 600, 16
00006BF2 1501200358021800    <1>     dw 115h,  800, 600, 24
00006BFA 1701000400031000    <1>     dw 117h, 1024, 768, 16
00006C02 1801000400031800    <1>     dw 118h, 1024, 768, 24
                             <1>
                             <1> ;/* BOCHS/PLEX86 'own' mode numbers */
00006C0A 40014001C8002000    <1>     dw 140h,  320,  200, 32
00006C12 4101800290012000    <1>     dw 141h,  640,  400, 32
00006C1A 42018002E0012000    <1>     dw 142h,  640,  480, 32
00006C22 4301200358022000    <1>     dw 143h,  800,  600, 32
00006C2A 4401000400032000    <1>     dw 144h, 1024,  768, 32
00006C32 46014001C8000800    <1>     dw 146h,  320,  200,  8
00006C3A 8D010005D0021000    <1>     dw 18Dh, 1280,  720, 16
00006C42 8E010005D0021800    <1>     dw 18Eh, 1280,  720, 24
00006C4A 8F010005D0022000    <1>     dw 18Fh, 1280,  720, 32
00006C52 9001800738041000    <1>     dw 190h, 1920, 1080, 16
00006C5A 9101800738041800    <1>     dw 191h, 1920, 1080, 24
00006C62 9201800738042000    <1>     dw 192h, 1920, 1080, 32
                             <1>
                             <1> end_of_b_vbe_modes:
                             <1>
                             <1> MA1 equ VBE_MODE_ATTRIBUTE_SUPPORTED
                             <1> MA2 equ VBE_MODE_ATTRIBUTE_EXTENDED_INFO_AVAILABLE
                             <1> MA3 equ VBE_MODE_ATTRIBUTE_COLOR_MODE
                             <1> MA4 equ VBE_MODE_ATTRIBUTE_LINEAR_FRAME_BUFFER_MODE
                             <1> MA5 equ     VBE_MODE_ATTRIBUTE_GRAPHICS_MODE
                             <1>
                             <1> MODE_ATTRIBUTES equ MA1|MA2|MA3|MA4|MA5
                             <1>
                             <1> WA1 equ VBE_WINDOW_ATTRIBUTE_RELOCATABLE
                             <1> WA2 equ     VBE_WINDOW_ATTRIBUTE_READABLE
                             <1> WA3 equ VBE_WINDOW_ATTRIBUTE_WRITEABLE
                             <1>
                             <1> WINA_ATTRIBUTES equ WA1|WA2|WA3
                             <1>
                             <1> ; 24/11/2020
                             <1>
                             <1> %if 0
                             <1>
                             <1> MODE_INFO_LIST:
                             <1>
                             <1> ; 24/11/2020
                             <1> ; '%if 0' disables 24 mode info tables here, until %endif
                             <1> ; ('set_mode_info_list' will set only 1 list for selected mode)
                             <1> ; (Purpose: To save about 1 KB kernel size by removing fixed data)
                             <1>
                             <1>                     dw      0100h ; 640x400x8
                             <1> ModeAttributes1:    dw      MODE_ATTRIBUTES
                             <1> WinAAttributes1:    db      WINA_ATTRIBUTES
                             <1> WinBAttributes1:    db      0
                             <1> WinGranularity1:    dw      VBE_DISPI_BANK_SIZE_KB
                             <1> WinSize1:           dw      VBE_DISPI_BANK_SIZE_KB
                             <1> WinASegment1:               dw      VGAMEM_GRAPH
                             <1> WinBSegment1:               dw      0000h
                             <1> WinFuncPtr1:                dd      0
                             <1> BytesPerScanLine1:          dw      640
                             <1> XResolution1:               dw      640
                             <1> YResolution1:               dw      400
                             <1> XCharSize1:         db      8
                             <1> YCharSize1:         db      16
                             <1> NumberOfPlanes1:    db      1
                             <1> BitsPerPixel1:              db      8
                             <1> NumberOfBanks1:             db      4
                             <1> MemoryModel1:               db      VBE_MEMORYMODEL_PACKED_PIXEL
                             <1> BankSize1:          db      0
                             <1> NumberOfImagePages1:        db      64
                             <1> Reserved_page1:             db      0
                             <1> RedMaskSize1:               db      0
                             <1> RedFieldPosition1:          db      0
                             <1> GreenMaskSize1:             db      0
                             <1> GreenFieldPosition1:        db      0
                             <1> BlueMaskSize1:              db      0
                             <1> BlueFieldPosition1:         db      0
                             <1> RsvdMaskSize1:              db      0
                             <1> RsvdFieldPosition1:         db      0
                             <1> DirectColorModeInfo1:       db      0
                             <1> PhysBasePtr1:               dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS
                             <1> OffScreenMemOffset1:        dd      0
                             <1> OffScreenMemSize1:          dw      0
                             <1> LinBytesPerScanLine1:       dw      640
                             <1> BnkNumberOfPages1:          db      0
                             <1> LinNumberOfPages1:          db      0
                             <1> LinRedMaskSize1:    db      0
                             <1> LinRedFieldPosition1:       db      0
                             <1> LinGreenMaskSize1:          db      0
                             <1> LinGreenFieldPosition1: db 0
                             <1> LinBlueMaskSize1:   db      0
                             <1> LinBlueFieldPosition1:      db      0
                             <1> LinRsvdMaskSize1:   db      0
                             <1> LinRsvdFieldPosition1:      db      0
                             <1> MaxPixelClock1:             dd      0
                             <1>
                             <1>                     dw      0101h ; 640x480x8
                             <1> ModeAttributes2:    dw      MODE_ATTRIBUTES
```

```
                        <1> WinAAttributes2:   db      WINA_ATTRIBUTES
                        <1> WinBAttributes2:   db      0
                        <1> WinGranularity2:   dw      VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize2:          dw      VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment2:              dw      VGAMEM_GRAPH
                        <1> WinBSegment2:              dw      0000h
                        <1> WinFuncPtr2:               dd      0
                        <1> BytesPerScanLine2: dw      640
                        <1> XResolution2:              dw      640
                        <1> YResolution2:              dw      480
                        <1> XCharSize2:        db      8
                        <1> YCharSize2:        db      16
                        <1> NumberOfPlanes2:   db      1
                        <1> BitsPerPixel2:             db      8
                        <1> NumberOfBanks2:            db      5
                        <1> MemoryModel2:              db      VBE_MEMORYMODEL_PACKED_PIXEL
                        <1> BankSize2:         db      0
                        <1> NumberOfImagePages2:       db      53
                        <1> Reserved_page2:            db      0
                        <1> RedMaskSize2:              db      0
                        <1> RedFieldPosition2: db      0
                        <1> GreenMaskSize2:            db      0
                        <1> GreenFieldPosition2:       db      0
                        <1> BlueMaskSize2:             db      0
                        <1> BlueFieldPosition2:        db      0
                        <1> RsvdMaskSize2:             db      0
                        <1> RsvdFieldPosition2:        db      0
                        <1> DirectColorModeInfo2:      db      0
                        <1> PhysBasePtr2:              dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS
                        <1> OffScreenMemOffset2:       dd      0
                        <1> OffScreenMemSize2: dw      0
                        <1> LinBytesPerScanLine2:      dw      640
                        <1> BnkNumberOfPages2:         db      0
                        <1> LinNumberOfPages2:         db      0
                        <1> LinRedMaskSize2:   db      0
                        <1> LinRedFieldPosition2:      db      0
                        <1> LinGreenMaskSize2:         db      0
                        <1> LinGreenFieldPosition2: db 0
                        <1> LinBlueMaskSize2:  db      0
                        <1> LinBlueFieldPosition2:     db      0
                        <1> LinRsvdMaskSize2:  db      0
                        <1> LinRsvdFieldPosition2:     db      0
                        <1> MaxPixelClock2:            dd      0
                        <1>
                        <1>                    dw      0103h ; 800x600x8
                        <1> ModeAttributes3:   dw      MODE_ATTRIBUTES
                        <1> WinAAttributes3:   db      WINA_ATTRIBUTES
                        <1> WinBAttributes3:   db      0
                        <1> WinGranularity3:   dw      VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize3:          dw      VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment3:              dw      VGAMEM_GRAPH
                        <1> WinBSegment3:              dw      0000h
                        <1> WinFuncPtr3:               dd      0
                        <1> BytesPerScanLine3: dw      800
                        <1> XResolution3:              dw      800
                        <1> YResolution3:              dw      600
                        <1> XCharSize3:        db      8
                        <1> YCharSize3:        db      16
                        <1> NumberOfPlanes3:   db      1
                        <1> BitsPerPixel3:             db      8
                        <1> NumberOfBanks3:            db      8
                        <1> MemoryModel3:              db      VBE_MEMORYMODEL_PACKED_PIXEL
                        <1> BankSize3:         db      0
                        <1> NumberOfImagePages3:       db      33
                        <1> Reserved_page3:            db      0
                        <1> RedMaskSize3:              db      0
                        <1> RedFieldPosition3: db      0
                        <1> GreenMaskSize3:            db      0
                        <1> GreenFieldPosition3:       db      0
                        <1> BlueMaskSize3:             db      0
                        <1> BlueFieldPosition3:        db      0
                        <1> RsvdMaskSize3:             db      0
                        <1> RsvdFieldPosition3:        db      0
                        <1> DirectColorModeInfo3:      db      0
                        <1> PhysBasePtr3:              dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS
                        <1> OffScreenMemOffset3:       dd      0
                        <1> OffScreenMemSize3: dw      0
                        <1> LinBytesPerScanLine3:      dw      800
                        <1> BnkNumberOfPages3:         db      0
                        <1> LinNumberOfPages3:         db      0
                        <1> LinRedMaskSize3:   db      0
                        <1> LinRedFieldPosition3:      db      0
                        <1> LinGreenMaskSize3:         db      0
                        <1> LinGreenFieldPosition:     db      0
                        <1> LinBlueMaskSize:   db      0
                        <1> LinBlueFieldPosition:      db      0
                        <1> LinRsvdMaskSize:   db      0
                        <1> LinRsvdFieldPosition:      db      0
                        <1> MaxPixelClock:             dd      0
                        <1>
                        <1>                    dw      0105h ; 1024x768x8
                        <1> ModeAttributes4:   dw      MODE_ATTRIBUTES
                        <1> WinAAttributes4:   db      WINA_ATTRIBUTES
                        <1> WinBAttributes4:   db      0
                        <1> WinGranularity4:   dw      VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize4:          dw      VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment4:              dw      VGAMEM_GRAPH
                        <1> WinBSegment4:              dw      0000h
                        <1> WinFuncPtr4:               dd      0
                        <1> BytesPerScanLine4: dw      1024
                        <1> XResolution4:              dw      1024
                        <1> YResolution4:              dw      768
                        <1> XCharSize4:        db      8
                        <1> YCharSize4:        db      16
                        <1> NumberOfPlanes4:   db      1
                        <1> BitsPerPixel4:             db      8
                        <1> NumberOfBanks4:            db      12
                        <1> MemoryModel4:              db      VBE_MEMORYMODEL_PACKED_PIXEL
                        <1> BankSize4:         db      0
                        <1> NumberOfImagePages4:       db      20
                        <1> Reserved_page4:            db      0
                        <1> RedMaskSize4:              db      0
                        <1> RedFieldPosition4: db      0
                        <1> GreenMaskSize4:            db      0
                        <1> GreenFieldPosition4:       db      0
                        <1> BlueMaskSize4:             db      0
                        <1> BlueFieldPosition4:        db      0
                        <1> RsvdMaskSize4:             db      0
                        <1> RsvdFieldPosition4:        db      0
                        <1> DirectColorModeInfo4:      db      0
                        <1> PhysBasePtr4:              dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS
                        <1> OffScreenMemOffset4:       dd      0
                        <1> OffScreenMemSize4: dw      0
                        <1> LinBytesPerScanLine4:      dw      1024
```

```
598                             <1> BnkNumberOfPages4:        db      0
599                             <1> LinNumberOfPages4:        db      0
600                             <1> LinRedMaskSize4:   db     0
601                             <1> LinRedFieldPosition4:     db      0
602                             <1> LinGreenMaskSize4:        db      0
603                             <1> LinGreenFieldPosition4: db 0
604                             <1> LinBlueMaskSize4:  db     0
605                             <1> LinBlueFieldPosition4:    db      0
606                             <1> LinRsvdMaskSize4:  db     0
607                             <1> LinRsvdFieldPosition4:    db      0
608                             <1> MaxPixelClock4:           dd      0
609                             <1>
610                             <1>                     dw      010Eh ; 320x200x16
611                             <1> ModeAttributes5:   dw      MODE_ATTRIBUTES
612                             <1> WinAAttributes5:   db      WINA_ATTRIBUTES
613                             <1> WinBAttributes5:   db      0
614                             <1> WinGranularity5:   dw      VBE_DISPI_BANK_SIZE_KB
615                             <1> WinSize5:          dw      VBE_DISPI_BANK_SIZE_KB
616                             <1> WinASegment5:              dw      VGAMEM_GRAPH
617                             <1> WinBSegment5:              dw      0000h
618                             <1> WinFuncPtr5:               dd      0
619                             <1> BytesPerScanLine5: dw      640
620                             <1> XResolution5:              dw      320
621                             <1> YResolution5:              dw      200
622                             <1> XCharSize5:        db      8
623                             <1> YCharSize5:        db      16
624                             <1> NumberOfPlanes5:   db      1
625                             <1> BitsPerPixel5:             db      16
626                             <1> NumberOfBanks5:            db      2
627                             <1> MemoryModel5:              db      VBE_MEMORYMODEL_DIRECT_COLOR
628                             <1> BankSize5:         db      0
629                             <1> NumberOfImagePages5:       db      130
630                             <1> Reserved_page5:            db      0
631                             <1> RedMaskSize5:              db      5
632                             <1> RedFieldPosition5: db      11
633                             <1> GreenMaskSize5:            db      6
634                             <1> GreenFieldPosition5:       db      5
635                             <1> BlueMaskSize5:             db      5
636                             <1> BlueFieldPosition5:        db      0
637                             <1> RsvdMaskSize5:             db      0
638                             <1> RsvdFieldPosition5:        db      0
639                             <1> DirectColorModeInfo5:      db      0
640                             <1> PhysBasePtr5:              dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS
641                             <1> OffScreenMemOffset5:       dd      0
642                             <1> OffScreenMemSize5: dw      0
643                             <1> LinBytesPerScanLine5:      dw      640
644                             <1> BnkNumberOfPages5:         db      0
645                             <1> LinNumberOfPages5:         db      0
646                             <1> LinRedMaskSize5:   db      5
647                             <1> LinRedFieldPosition5:      db      11
648                             <1> LinGreenMaskSize5: db      6
649                             <1> LinGreenFieldPosition5:    db      5
650                             <1> LinBlueMaskSize5:  db      5
651                             <1> LinBlueFieldPosition5:     db      0
652                             <1> LinRsvdMaskSize5:  db      0
653                             <1> LinRsvdFieldPosition5:     db      0
654                             <1> MaxPixelClock5:            dd      0
655                             <1>
656                             <1>                     dw      010Fh ; 320x200x24
657                             <1> ModeAttributes6:   dw      MODE_ATTRIBUTES
658                             <1> WinAAttributes6:   db      WINA_ATTRIBUTES
659                             <1> WinBAttributes6:   db      0
660                             <1> WinGranularity6:   dw      VBE_DISPI_BANK_SIZE_KB
661                             <1> WinSize6:          dw      VBE_DISPI_BANK_SIZE_KB
662                             <1> WinASegment6:              dw      VGAMEM_GRAPH
663                             <1> WinBSegment6:              dw      0000h
664                             <1> WinFuncPtr6:               dd      0
665                             <1> BytesPerScanLine6: dw      960
666                             <1> XResolution6:              dw      320
667                             <1> YResolution6:              dw      200
668                             <1> XCharSize6:        db      8
669                             <1> YCharSize6:        db      16
670                             <1> NumberOfPlanes6:   db      1
671                             <1> BitsPerPixel6:             db      24
672                             <1> NumberOfBanks6:            db      3
673                             <1> MemoryModel6:              db      VBE_MEMORYMODEL_DIRECT_COLOR
674                             <1> BankSize6:         db      0
675                             <1> NumberOfImagePages6:       db      86
676                             <1> Reserved_page6:            db      0
677                             <1> RedMaskSize6:              db      8
678                             <1> RedFieldPosition6: db      16
679                             <1> GreenMaskSize6:            db      8
680                             <1> GreenFieldPosition6:       db      8
681                             <1> BlueMaskSize6:             db      8
682                             <1> BlueFieldPosition6:        db      0
683                             <1> RsvdMaskSize6:             db      0
684                             <1> RsvdFieldPosition6:        db      0
685                             <1> DirectColorModeInfo6:      db      0
686                             <1> PhysBasePtr6:              dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS
687                             <1> OffScreenMemOffset6:       dd      0
688                             <1> OffScreenMemSize6: dw      0
689                             <1> LinBytesPerScanLine6:      dw      960
690                             <1> BnkNumberOfPages6:         db      0
691                             <1> LinNumberOfPages6:         db      0
692                             <1> LinRedMaskSize6:   db      8
693                             <1> LinRedFieldPosition6:      db      16
694                             <1> LinGreenMaskSize6: db      8
695                             <1> LinGreenFieldPosition6:    db      8
696                             <1> LinBlueMaskSize6:  db      8
697                             <1> LinBlueFieldPosition6:     db      0
698                             <1> LinRsvdMaskSize6:  db      0
699                             <1> LinRsvdFieldPosition6:     db      0
700                             <1> MaxPixelClock6:            dd      0
701                             <1>
702                             <1>                     dw      0111h ; 640x480x16
703                             <1> ModeAttributes7:   dw      MODE_ATTRIBUTES
704                             <1> WinAAttributes7:   db      WINA_ATTRIBUTES
705                             <1> WinBAttributes7:   db      0
706                             <1> WinGranularity7:   dw      VBE_DISPI_BANK_SIZE_KB
707                             <1> WinSize7:          dw      VBE_DISPI_BANK_SIZE_KB
708                             <1> WinASegment7:              dw      VGAMEM_GRAPH
709                             <1> WinBSegment7:              dw      0000h
710                             <1> WinFuncPtr7:               dd      0
711                             <1> BytesPerScanLine7: dw      1280
712                             <1> XResolution7:              dw      640
713                             <1> YResolution7:              dw      480
714                             <1> XCharSize7:        db      8
715                             <1> YCharSize7:        db      16
716                             <1> NumberOfPlanes7:   db      1
717                             <1> BitsPerPixel7:             db      16
718                             <1> NumberOfBanks7:            db      10
719                             <1> MemoryModel7:              db      VBE_MEMORYMODEL_DIRECT_COLOR
720                             <1> BankSize7:         db      0
721                             <1> NumberOfImagePages7:       db      26
```

```
722                                     <1> Reserved_page7:           db      0
723                                     <1> RedMaskSize7:             db      5
724                                     <1> RedFieldPosition7: db     11
725                                     <1> GreenMaskSize7:           db      6
726                                     <1> GreenFieldPosition7:      db      5
727                                     <1> BlueMaskSize7:            db      5
728                                     <1> BlueFieldPosition7:       db      0
729                                     <1> RsvdMaskSize7:            db      0
730                                     <1> RsvdFieldPosition7:       db      0
731                                     <1> DirectColorModeInfo7:     db      0
732                                     <1> PhysBasePtr7:             dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS,
733                                     <1> OffScreenMemOffset7:      dd      0
734                                     <1> OffScreenMemSize7: dw     0
735                                     <1> LinBytesPerScanLine7:     dw      1280
736                                     <1> BnkNumberOfPages7:        db      0
737                                     <1> LinNumberOfPages7:        db      0
738                                     <1> LinRedMaskSize7:   db     5
739                                     <1> LinRedFieldPosition7:     db      11
740                                     <1> LinGreenMaskSize7: db     6
741                                     <1> LinGreenFieldPosition7:   db      5
742                                     <1> LinBlueMaskSize7:  db     5
743                                     <1> LinBlueFieldPosition7:    db      0
744                                     <1> LinRsvdMaskSize7:  db     0
745                                     <1> LinRsvdFieldPosition7:    db      0
746                                     <1> MaxPixelClock7:           dd      0
747                                     <1>
748                                     <1>                     dw      0112h ; 640x480x24
749                                     <1> ModeAttributes8:    dw      MODE_ATTRIBUTES
750                                     <1> WinAAttributes8:    db      WINA_ATTRIBUTES
751                                     <1> WinBAttributes8:    db      0
752                                     <1> WinGranularity8:    dw      VBE_DISPI_BANK_SIZE_KB
753                                     <1> WinSize8:           dw      VBE_DISPI_BANK_SIZE_KB
754                                     <1> WinASegment8:             dw      VGAMEM_GRAPH
755                                     <1> WinBSegment8:             dw      0000h
756                                     <1> WinFuncPtr8:              dd      0
757                                     <1> BytesPerScanLine8: dw     1920
758                                     <1> XResolution8:             dw      640
759                                     <1> YResolution8:             dw      480
760                                     <1> XCharSize8:         db      8
761                                     <1> YCharSize8:         db      16
762                                     <1> NumberOfPlanes8:    db      1
763                                     <1> BitsPerPixel8:            db      24
764                                     <1> NumberOfBanks8:           db      15
765                                     <1> MemoryModel8:             db      VBE_MEMORYMODEL_DIRECT_COLOR
766                                     <1> BankSize8:          db      0
767                                     <1> NumberOfImagePages8:      db      17
768                                     <1> Reserved_page8:           db      0
769                                     <1> RedMaskSize8:             db      8
770                                     <1> RedFieldPosition8: db     16
771                                     <1> GreenMaskSize8:           db      8
772                                     <1> GreenFieldPosition8:      db      8
773                                     <1> BlueMaskSize8:            db      8
774                                     <1> BlueFieldPosition8:       db      0
775                                     <1> RsvdMaskSize8:            db      0
776                                     <1> RsvdFieldPosition8:       db      0
777                                     <1> DirectColorModeInfo8:     db      0
778                                     <1> PhysBasePtr8:             dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS
779                                     <1> OffScreenMemOffset8:      dd      0
780                                     <1> OffScreenMemSize8: dw     0
781                                     <1> LinBytesPerScanLine8:     dw      1920
782                                     <1> BnkNumberOfPages8:        db      0
783                                     <1> LinNumberOfPages8:        db      0
784                                     <1> LinRedMaskSize8:   db     8
785                                     <1> LinRedFieldPosition8:     db      16
786                                     <1> LinGreenMaskSize8: db     8
787                                     <1> LinGreenFieldPosition8:   db      8
788                                     <1> LinBlueMaskSize8:  db     8
789                                     <1> LinBlueFieldPosition8:    db      0
790                                     <1> LinRsvdMaskSize8:  db     0
791                                     <1> LinRsvdFieldPosition8:    db      0
792                                     <1> MaxPixelClock8:           dd      0
793                                     <1>
794                                     <1>                     dw      0114h ; 800x600x16
795                                     <1> ModeAttributes9:    dw      MODE_ATTRIBUTES
796                                     <1> WinAAttributes9:    db      WINA_ATTRIBUTES
797                                     <1> WinBAttributes9:    db      0
798                                     <1> WinGranularity9:    dw      VBE_DISPI_BANK_SIZE_KB
799                                     <1> WinSize9:           dw      VBE_DISPI_BANK_SIZE_KB
800                                     <1> WinASegment9:             dw      VGAMEM_GRAPH
801                                     <1> WinBSegment9:             dw      0000h
802                                     <1> WinFuncPtr9:              dd      0
803                                     <1> BytesPerScanLine9: dw     1600
804                                     <1> XResolution9:             dw      800
805                                     <1> YResolution9:             dw      600
806                                     <1> XCharSize9:         db      8
807                                     <1> YCharSize9:         db      16
808                                     <1> NumberOfPlanes9:    db      1
809                                     <1> BitsPerPixel9:            db      16
810                                     <1> NumberOfBanks9:           db      15
811                                     <1> MemoryModel9:             db      VBE_MEMORYMODEL_DIRECT_COLOR
812                                     <1> BankSize9:          db      0
813                                     <1> NumberOfImagePages9:      db      16
814                                     <1> Reserved_page9:           db      0
815                                     <1> RedMaskSize9:             db      5
816                                     <1> RedFieldPosition9: db     11
817                                     <1> GreenMaskSize9:           db      6
818                                     <1> GreenFieldPosition9:      db      5
819                                     <1> BlueMaskSize9:            db      5
820                                     <1> BlueFieldPosition9:       db      0
821                                     <1> RsvdMaskSize9:            db      0
822                                     <1> RsvdFieldPosition9:       db      0
823                                     <1> DirectColorModeInfo9:     db      0
824                                     <1> PhysBasePtr9:             dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS
825                                     <1> OffScreenMemOffset9:      dd      0
826                                     <1> OffScreenMemSize9: dw     0
827                                     <1> LinBytesPerScanLine9:     dw      1600
828                                     <1> BnkNumberOfPages9:        db      0
829                                     <1> LinNumberOfPages9:        db      0
830                                     <1> LinRedMaskSize9:   db     5
831                                     <1> LinRedFieldPosition9:     db      11
832                                     <1> LinGreenMaskSize9: db     6
833                                     <1> LinGreenFieldPosition9:   db      5
834                                     <1> LinBlueMaskSize9:  db     5
835                                     <1> LinBlueFieldPosition9:    db      0
836                                     <1> LinRsvdMaskSize9:  db     0
837                                     <1> LinRsvdFieldPosition9:    db      0
838                                     <1> MaxPixelClock9:           dd      0
839                                     <1>
840                                     <1>                     dw      0115h ; 800x600x24
841                                     <1> ModeAttributes10:   dw      MODE_ATTRIBUTES
842                                     <1> WinAAttributes10:   db      WINA_ATTRIBUTES
843                                     <1> WinBAttributes10:   db      0
844                                     <1> WinGranularity10:   dw      VBE_DISPI_BANK_SIZE_KB
845                                     <1> WinSize10:          dw      VBE_DISPI_BANK_SIZE_KB
```

```
                        <1> WinASegment10:            dw      VGAMEM_GRAPH
                        <1> WinBSegment10:            dw      0000h
                        <1> WinFuncPtr10:             dd      0
                        <1> BytesPerScanLine10:       dw      2400
                        <1> XResolution10:            dw      800
                        <1> YResolution10:            dw      600
                        <1> XCharSize10:              db      8
                        <1> YCharSize10:              db      16
                        <1> NumberOfPlanes10:  db     1
                        <1> BitsPerPixel10:           db      24
                        <1> NumberOfBanks10:   db     22
                        <1> MemoryModel10:            db      VBE_MEMORYMODEL_DIRECT_COLOR
                        <1> BankSize10:        db     0
                        <1> NumberOfImagePages10:     db      10
                        <1> Reserved_page10:   db     0
                        <1> RedMaskSize10:            db      8
                        <1> RedFieldPosition10:       db      16
                        <1> GreenMaskSize10:   db     8
                        <1> GreenFieldPosition10:     db      8
                        <1> BlueMaskSize10:           db      8
                        <1> BlueFieldPosition10:      db      0
                        <1> RsvdMaskSize10:           db      0
                        <1> RsvdFieldPosition10:      db      0
                        <1> DirectColorModeInfo10:    db      0
                        <1> PhysBasePtr10:            dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS
                        <1> OffScreenMemOffset10:     dd      0
                        <1> OffScreenMemSize10:       dw      0
                        <1> LinBytesPerScanLine10:    dw      2400
                        <1> BnkNumberOfPages10:       db      0
                        <1> LinNumberOfPages10:       db      0
                        <1> LinRedMaskSize10:  db     8
                        <1> LinRedFieldPosition10:    db      16
                        <1> LinGreenMaskSize10:       db      8
                        <1> LinGreenFieldPosition10:db 8
                        <1> LinBlueMaskSize10: db     8
                        <1> LinBlueFieldPosition10:   db      0
                        <1> LinRsvdMaskSize10: db     0
                        <1> LinRsvdFieldPosition10:   db      0
                        <1> MaxPixelClock10:   dd     0
                        <1>
                        <1>                           dw      0117h ; 1024x768x16
                        <1> ModeAttributes11:  dw     MODE_ATTRIBUTES
                        <1> WinAAttributes11:  db     WINA_ATTRIBUTES
                        <1> WinBAttributes11:  db     0
                        <1> WinGranularity11:  dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize11:         dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment11:            dw      VGAMEM_GRAPH
                        <1> WinBSegment11:            dw      0000h
                        <1> WinFuncPtr11:             dd      0
                        <1> BytesPerScanLine11:       dw      2048
                        <1> XResolution11:            dw      1024
                        <1> YResolution11:            dw      768
                        <1> XCharSize11:              db      8
                        <1> YCharSize11:              db      16
                        <1> NumberOfPlanes11:  db     1
                        <1> BitsPerPixel11:           db      16
                        <1> NumberOfBanks11:   db     24
                        <1> MemoryModel11:            db      VBE_MEMORYMODEL_DIRECT_COLOR
                        <1> BankSize11:        db     0
                        <1> NumberOfImagePages11:     db      9
                        <1> Reserved_page11:   db     0
                        <1> RedMaskSize11:            db      5
                        <1> RedFieldPosition11:       db      11
                        <1> GreenMaskSize11:   db     6
                        <1> GreenFieldPosition11:     db      5
                        <1> BlueMaskSize11:           db      5
                        <1> BlueFieldPosition11:      db      0
                        <1> RsvdMaskSize11:           db      0
                        <1> RsvdFieldPosition11:      db      0
                        <1> DirectColorModeInfo11:    db      0
                        <1> PhysBasePtr11:            dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS,
                        <1> OffScreenMemOffset11:     dd      0
                        <1> OffScreenMemSize11:       dw      0
                        <1> LinBytesPerScanLine11:    dw      2048
                        <1> BnkNumberOfPages11:       db      0
                        <1> LinNumberOfPages11:       db      0
                        <1> LinRedMaskSize11:  db     5
                        <1> LinRedFieldPosition11:    db      11
                        <1> LinGreenMaskSize11:       db      6
                        <1> LinGreenFieldPosition11:db 5
                        <1> LinBlueMaskSize11: db     5
                        <1> LinBlueFieldPosition11:   db      0
                        <1> LinRsvdMaskSize11: db     0
                        <1> LinRsvdFieldPosition11:   db      0
                        <1> MaxPixelClock11:   dd     0
                        <1>
                        <1>                           dw      0118h ; 1024x768x24
                        <1> ModeAttributes12:  dw     MODE_ATTRIBUTES
                        <1> WinAAttributes12:  db     WINA_ATTRIBUTES
                        <1> WinBAttributes12:  db     0
                        <1> WinGranularity12:  dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize12:         dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment12:            dw      VGAMEM_GRAPH
                        <1> WinBSegment12:            dw      0000h
                        <1> WinFuncPtr12:             dd      0
                        <1> BytesPerScanLine12:       dw      3072
                        <1> XResolution12:            dw      1024
                        <1> YResolution12:            dw      768
                        <1> XCharSize12:              db      8
                        <1> YCharSize12:              db      16
                        <1> NumberOfPlanes12:  db     1
                        <1> BitsPerPixel12:           db      24
                        <1> NumberOfBanks12:   db     36
                        <1> MemoryModel12:            db      VBE_MEMORYMODEL_DIRECT_COLOR
                        <1> BankSize12:        db     0
                        <1> NumberOfImagePages12:     db      6
                        <1> Reserved_page12:   db     0
                        <1> RedMaskSize12:            db      8
                        <1> RedFieldPosition12:       db      16
                        <1> GreenMaskSize12:   db     8
                        <1> GreenFieldPosition12:     db      8
                        <1> BlueMaskSize12:           db      8
                        <1> BlueFieldPosition12:      db      0
                        <1> RsvdMaskSize12:           db      0
                        <1> RsvdFieldPosition12:      db      0
                        <1> DirectColorModeInfo12:    db      0
                        <1> PhysBasePtr12:            dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS
                        <1> OffScreenMemOffset12:     dd      0
                        <1> OffScreenMemSize12:       dw      0
                        <1> LinBytesPerScanLine12:    dw      3072
                        <1> BnkNumberOfPages12:       db      0
                        <1> LinNumberOfPages12:       db      0
                        <1> LinRedMaskSize12:  db     8
                        <1> LinRedFieldPosition12:    db      16
```

```
                        <1> LinGreenMaskSize12:       db      8
                        <1> LinGreenFieldPosition12:db 8
                        <1> LinBlueMaskSize12: db     8
                        <1> LinBlueFieldPosition12:   db      0
                        <1> LinRsvdMaskSize12: db     0
                        <1> LinRsvdFieldPosition12:   db      0
                        <1> MaxPixelClock12:   dd     0
                        <1>
                        <1>                    dw     0140h ; 320x200x32
                        <1> ModeAttributes13:  dw     MODE_ATTRIBUTES
                        <1> WinAAttributes13:  db     WINA_ATTRIBUTES
                        <1> WinBAttributes13:  db     0
                        <1> WinGranularity13:  dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize13:         dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment13:             dw     VGAMEM_GRAPH
                        <1> WinBSegment13:             dw     0000h
                        <1> WinFuncPtr13:              dd     0
                        <1> BytesPerScanLine13:        dw     1280
                        <1> XResolution13:             dw     320
                        <1> YResolution13:             dw     200
                        <1> XCharSize13:               db     8
                        <1> YCharSize13:               db     16
                        <1> NumberOfPlanes13:  db     1
                        <1> BitsPerPixel13:            db     32
                        <1> NumberOfBanks13:   db     4
                        <1> MemoryModel13:             db     VBE_MEMORYMODEL_DIRECT_COLOR
                        <1> BankSize13:        db     0
                        <1> NumberOfImagePages13:      db     64
                        <1> Reserved_page13:   db     0
                        <1> RedMaskSize13:             db     8
                        <1> RedFieldPosition13:        db     16
                        <1> GreenMaskSize13:   db     8
                        <1> GreenFieldPosition13:      db     8
                        <1> BlueMaskSize13:            db     8
                        <1> BlueFieldPosition13:       db     0
                        <1> RsvdMaskSize13:            db     8
                        <1> RsvdFieldPosition13:       db     24
                        <1> DirectColorModeInfo13:     db     VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
                        <1> PhysBasePtr13:             dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
                        <1> OffScreenMemOffset13:      dd     0
                        <1> OffScreenMemSize13:        dw     0
                        <1> LinBytesPerScanLine13:     dw     1280
                        <1> BnkNumberOfPages13:        db     0
                        <1> LinNumberOfPages13:        db     0
                        <1> LinRedMaskSize13:  db     8
                        <1> LinRedFieldPosition13:     db     16
                        <1> LinGreenMaskSize13:        db     8
                        <1> LinGreenFieldPosition13:db 8
                        <1> LinBlueMaskSize13: db     8
                        <1> LinBlueFieldPosition13:   db      0
                        <1> LinRsvdMaskSize13: db     8
                        <1> LinRsvdFieldPosition13:   db      24
                        <1> MaxPixelClock13:   dd     0
                        <1>
                        <1>                    dw     0141h ; 640x400x32
                        <1> ModeAttributes14:  dw     MODE_ATTRIBUTES
                        <1> WinAAttributes14:  db     WINA_ATTRIBUTES
                        <1> WinBAttributes14:  db     0
                        <1> WinGranularity14:  dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize14:         dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment14:             dw     VGAMEM_GRAPH
                        <1> WinBSegment14:             dw     0000h
                        <1> WinFuncPtr14:              dd     0
                        <1> BytesPerScanLine14:        dw     2560
                        <1> XResolution14:             dw     640
                        <1> YResolution14:             dw     400
                        <1> XCharSize14:               db     8
                        <1> YCharSize14:               db     16
                        <1> NumberOfPlanes14:  db     1
                        <1> BitsPerPixel14:            db     32
                        <1> NumberOfBanks14:   db     16
                        <1> MemoryModel14:             db     VBE_MEMORYMODEL_DIRECT_COLOR
                        <1> BankSize14:        db     0
                        <1> NumberOfImagePages14:      db     15
                        <1> Reserved_page14:   db     0
                        <1> RedMaskSize14:             db     8
                        <1> RedFieldPosition14:        db     16
                        <1> GreenMaskSize14:   db     8
                        <1> GreenFieldPosition14:      db     8
                        <1> BlueMaskSize14:            db     8
                        <1> BlueFieldPosition14:       db     0
                        <1> RsvdMaskSize14:            db     8
                        <1> RsvdFieldPosition14:       db     24
                        <1> DirectColorModeInfo14:     db     VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
                        <1> PhysBasePtr14:             dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
                        <1> OffScreenMemOffset14:      dd     0
                        <1> OffScreenMemSize14:        dw     0
                        <1> LinBytesPerScanLine14:     dw     2560
                        <1> BnkNumberOfPages14:        db     0
                        <1> LinNumberOfPages14:        db     0
                        <1> LinRedMaskSize14:  db     8
                        <1> LinRedFieldPosition14:     db     16
                        <1> LinGreenMaskSize14:        db     8
                        <1> LinGreenFieldPosition14:db 8
                        <1> LinBlueMaskSize14: db     8
                        <1> LinBlueFieldPosition14:   db      0
                        <1> LinRsvdMaskSize14: db     8
                        <1> LinRsvdFieldPosition14:   db      24
                        <1> MaxPixelClock14:   dd     0
                        <1>
                        <1>                    dw     0142 ; 640x480x32
                        <1> ModeAttributes15:  dw     MODE_ATTRIBUTES
                        <1> WinAAttributes15:  db     WINA_ATTRIBUTES
                        <1> WinBAttributes15:  db     0
                        <1> WinGranularity15:  dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize15:         dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment15:             dw     VGAMEM_GRAPH
                        <1> WinBSegment15:             dw     0000h
                        <1> WinFuncPtr15:              dd     0
                        <1> BytesPerScanLine15:        dw     2560
                        <1> XResolution15:             dw     640
                        <1> YResolution15:             dw     480
                        <1> XCharSize15:               db     8
                        <1> YCharSize15:               db     16
                        <1> NumberOfPlanes15:  db     1
                        <1> BitsPerPixel15:            db     32
                        <1> NumberOfBanks15:   db     19
                        <1> MemoryModel15:             db     VBE_MEMORYMODEL_DIRECT_COLOR
                        <1> BankSize15:        db     0
                        <1> NumberOfImagePages15:      db     12
                        <1> Reserved_page15:   db     0
                        <1> RedMaskSize15:             db     8
                        <1> RedFieldPosition15:        db     16
                        <1> GreenMaskSize15:   db     8
```

```
1094                            <1> GreenFieldPosition15:     db      8
1095                            <1> BlueMaskSize15:           db      8
1096                            <1> BlueFieldPosition15:      db      0
1097                            <1> RsvdMaskSize15:           db      8
1098                            <1> RsvdFieldPosition15:      db      24
1099                            <1> DirectColorModeInfo15:    db      VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE,
1100                            <1> PhysBasePtr15:            dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS,
1101                            <1> OffScreenMemOffset15:     dd      0
1102                            <1> OffScreenMemSize15:       dw      0
1103                            <1> LinBytesPerScanLine15:    dw      2560
1104                            <1> BnkNumberOfPages15:       db      0
1105                            <1> LinNumberOfPages15:       db      0
1106                            <1> LinRedMaskSize15:  db     8
1107                            <1> LinRedFieldPosition15:    db      16
1108                            <1> LinGreenMaskSize15:       db      8
1109                            <1> LinGreenFieldPosition15:db 8
1110                            <1> LinBlueMaskSize15: db     8
1111                            <1> LinBlueFieldPosition15:   db      0
1112                            <1> LinRsvdMaskSize15: db     8
1113                            <1> LinRsvdFieldPosition15:   db      24
1114                            <1> MaxPixelClock15:   dd     0
1115                            <1>
1116                            <1>                     dw      0143h ; 800x600x32
1117                            <1> ModeAttributes16:  dw      MODE_ATTRIBUTES
1118                            <1> WinAAttributes16:  db      WINA_ATTRIBUTES
1119                            <1> WinBAttributes16:  db      0
1120                            <1> WinGranularity16:  dw      VBE_DISPI_BANK_SIZE_KB
1121                            <1> WinSize16:         dw      VBE_DISPI_BANK_SIZE_KB
1122                            <1> WinASegment16:             dw      VGAMEM_GRAPH
1123                            <1> WinBSegment16:             dw      0000h
1124                            <1> WinFuncPtr16:              dd      0
1125                            <1> BytesPerScanLine16:        dw      3200
1126                            <1> XResolution16:             dw      800
1127                            <1> YResolution16:             dw      600
1128                            <1> XCharSize16:               db      8
1129                            <1> YCharSize16:               db      16
1130                            <1> NumberOfPlanes16:  db      1
1131                            <1> BitsPerPixel16:            db      32
1132                            <1> NumberOfBanks16:   db      30
1133                            <1> MemoryModel16:             db      VBE_MEMORYMODEL_DIRECT_COLOR
1134                            <1> BankSize16:        db      0
1135                            <1> NumberOfImagePages16:      db      7
1136                            <1> Reserved_page16:   db      0
1137                            <1> RedMaskSize16:             db      8
1138                            <1> RedFieldPosition16:        db      16
1139                            <1> GreenMaskSize16:   db      8
1140                            <1> GreenFieldPosition16:      db      8
1141                            <1> BlueMaskSize16:            db      8
1142                            <1> BlueFieldPosition16:       db      0
1143                            <1> RsvdMaskSize16:            db      8
1144                            <1> RsvdFieldPosition16:       db      24
1145                            <1> DirectColorModeInfo16:     db      VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE,
1146                            <1> PhysBasePtr16:             dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS,
1147                            <1> OffScreenMemOffset16:      dd      0
1148                            <1> OffScreenMemSize16:        dw      0
1149                            <1> LinBytesPerScanLine16:     dw      3200
1150                            <1> BnkNumberOfPages16:        db      0
1151                            <1> LinNumberOfPages16:        db      0
1152                            <1> LinRedMaskSize16:  db      8
1153                            <1> LinRedFieldPosition16:     db      16
1154                            <1> LinGreenMaskSize16:        db      8
1155                            <1> LinGreenFieldPosition16:db 8
1156                            <1> LinBlueMaskSize16: db      8
1157                            <1> LinBlueFieldPosition16:    db      0
1158                            <1> LinRsvdMaskSize16: db      8
1159                            <1> LinRsvdFieldPosition16:    db      24
1160                            <1> MaxPixelClock16:   dd      0
1161                            <1>
1162                            <1>                     dw      0144h ; 1024x768x32
1163                            <1> ModeAttributes17:  dw      MODE_ATTRIBUTES
1164                            <1> WinAAttributes17:  db      WINA_ATTRIBUTES
1165                            <1> WinBAttributes17:  db      0
1166                            <1> WinGranularity17:  dw      VBE_DISPI_BANK_SIZE_KB
1167                            <1> WinSize17:         dw      VBE_DISPI_BANK_SIZE_KB
1168                            <1> WinASegment17:             dw      VGAMEM_GRAPH
1169                            <1> WinBSegment17:             dw      0000h
1170                            <1> WinFuncPtr17:              dd      0
1171                            <1> BytesPerScanLine17:        dw      4096
1172                            <1> XResolution17:             dw      1024
1173                            <1> YResolution17:             dw      768
1174                            <1> XCharSize17:               db      8
1175                            <1> YCharSize17:               db      16
1176                            <1> NumberOfPlanes17:  db      1
1177                            <1> BitsPerPixel17:            db      32
1178                            <1> NumberOfBanks17:   db      48
1179                            <1> MemoryModel17:             db      VBE_MEMORYMODEL_DIRECT_COLOR
1180                            <1> BankSize17:        db      0
1181                            <1> NumberOfImagePages17:      db      4
1182                            <1> Reserved_page17:   db      0
1183                            <1> RedMaskSize17:             db      8
1184                            <1> RedFieldPosition17:        db      16
1185                            <1> GreenMaskSize17:   db      8
1186                            <1> GreenFieldPosition17:      db      8
1187                            <1> BlueMaskSize17:            db      8
1188                            <1> BlueFieldPosition17:       db      0
1189                            <1> RsvdMaskSize17:            db      8
1190                            <1> RsvdFieldPosition17:       db      24
1191                            <1> DirectColorModeInfo17:     db      VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
1192                            <1> PhysBasePtr17:             dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS
1193                            <1> OffScreenMemOffset17:      dd      0
1194                            <1> OffScreenMemSize17:        dw      0
1195                            <1> LinBytesPerScanLine17:     dw      4096
1196                            <1> BnkNumberOfPages17:        db      0
1197                            <1> LinNumberOfPages17:        db      0
1198                            <1> LinRedMaskSize17:  db      8
1199                            <1> LinRedFieldPosition17:     db      16
1200                            <1> LinGreenMaskSize17:        db      8
1201                            <1> LinGreenFieldPosition17:db 8
1202                            <1> LinBlueMaskSize17: db      8
1203                            <1> LinBlueFieldPosition17:    db      0
1204                            <1> LinRsvdMaskSize17: db      8
1205                            <1> LinRsvdFieldPosition17:    db      24
1206                            <1> MaxPixelClock17:   dd      0
1207                            <1>
1208                            <1>                     dw      0146h ; 320x200x8
1209                            <1> ModeAttributes18:  dw      MODE_ATTRIBUTES
1210                            <1> WinAAttributes18:  db      WINA_ATTRIBUTES
1211                            <1> WinBAttributes18:  db      0
1212                            <1> WinGranularity18:  dw      VBE_DISPI_BANK_SIZE_KB
1213                            <1> WinSize18:         dw      VBE_DISPI_BANK_SIZE_KB
1214                            <1> WinASegment18:             dw      VGAMEM_GRAPH
1215                            <1> WinBSegment18:             dw      0000h
1216                            <1> WinFuncPtr18:              dd      0
1217                            <1> BytesPerScanLine18:        dw      320
```

```
1218                            <1> XResolution18:            dw      320
1219                            <1> YResolution18:            dw      200
1220                            <1> XCharSize18:              db      8
1221                            <1> YCharSize18:              db      16
1222                            <1> NumberOfPlanes18:  db     1
1223                            <1> BitsPerPixel18:           db      8
1224                            <1> NumberOfBanks18:   db     1
1225                            <1> MemoryModel18:            db      VBE_MEMORYMODEL_PACKED_PIXEL
1226                            <1> BankSize18:        db     0
1227                            <1> NumberOfImagePages18:     db      255 ; 261 in vbetables.h (03/01/2020) !
1228                            <1> Reserved_page18:   db     0
1229                            <1> RedMaskSize18:            db      0
1230                            <1> RedFieldPosition18:       db      0
1231                            <1> GreenMaskSize18:   db     0
1232                            <1> GreenFieldPosition18:     db      0
1233                            <1> BlueMaskSize18:           db      0
1234                            <1> BlueFieldPosition18:      db      0
1235                            <1> RsvdMaskSize18:           db      0
1236                            <1> RsvdFieldPosition18:      db      0
1237                            <1> DirectColorModeInfo18:    db      0
1238                            <1> PhysBasePtr18:            dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS
1239                            <1> OffScreenMemOffset18:     dd      0
1240                            <1> OffScreenMemSize18:       dw      0
1241                            <1> LinBytesPerScanLine18:    dw      320
1242                            <1> BnkNumberOfPages18:       db      0
1243                            <1> LinNumberOfPages18:       db      0
1244                            <1> LinRedMaskSize18:  db     0
1245                            <1> LinRedFieldPosition18:    db      0
1246                            <1> LinGreenMaskSize18:       db      0
1247                            <1> LinGreenFieldPosition18:db 0
1248                            <1> LinBlueMaskSize18:        db      0
1249                            <1> LinBlueFieldPosition18: db 0
1250                            <1> LinRsvdMaskSize18:        db      0
1251                            <1> LinRsvdFieldPosition18: db 0
1252                            <1> MaxPixelClock18:   dd     0
1253                            <1>
1254                            <1>                    dw     018Dh ; 1280x720x16
1255                            <1> ModeAttributes19:  dw     MODE_ATTRIBUTES
1256                            <1> WinAAttributes19:  db     WINA_ATTRIBUTES
1257                            <1> WinBAttributes19:  db     0
1258                            <1> WinGranularity19:  dw     VBE_DISPI_BANK_SIZE_KB
1259                            <1> WinSize19:         dw     VBE_DISPI_BANK_SIZE_KB
1260                            <1> WinASegment19:            dw      VGAMEM_GRAPH
1261                            <1> WinBSegment19:            dw      0000h
1262                            <1> WinFuncPtr19:             dd      0
1263                            <1> BytesPerScanLine19:       dw      2560
1264                            <1> XResolution19:            dw      1280
1265                            <1> YResolution19:            dw      720
1266                            <1> XCharSize19:              db      8
1267                            <1> YCharSize19:              db      16
1268                            <1> NumberOfPlanes19:  db     1
1269                            <1> BitsPerPixel19:           db      16
1270                            <1> NumberOfBanks19:   db     29
1271                            <1> MemoryModel19:            db      VBE_MEMORYMODEL_DIRECT_COLOR
1272                            <1> BankSize19:        db     0
1273                            <1> NumberOfImagePages19:     db      8
1274                            <1> Reserved_page19:   db     0
1275                            <1> RedMaskSize19:            db      5
1276                            <1> RedFieldPosition19:       db      11
1277                            <1> GreenMaskSize19:   db     6
1278                            <1> GreenFieldPosition19:     db      5
1279                            <1> BlueMaskSize19:           db      5
1280                            <1> BlueFieldPosition19:      db      0
1281                            <1> RsvdMaskSize19:           db      0
1282                            <1> RsvdFieldPosition19:      db      0
1283                            <1> DirectColorModeInfo19:    db      0
1284                            <1> PhysBasePtr19:            dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS
1285                            <1> OffScreenMemOffset19:     dd      0
1286                            <1> OffScreenMemSize19:       dw      0
1287                            <1> LinBytesPerScanLine19:    dw      2560
1288                            <1> BnkNumberOfPages19:       db      0
1289                            <1> LinNumberOfPages19:       db      0
1290                            <1> LinRedMaskSize19:  db     5
1291                            <1> LinRedFieldPosition19:    db      11
1292                            <1> LinGreenMaskSize19:       db      6
1293                            <1> LinGreenFieldPosition19:db 5
1294                            <1> LinBlueMaskSize19: db     5
1295                            <1> LinBlueFieldPosition19:   db      0
1296                            <1> LinRsvdMaskSize19: db     0
1297                            <1> LinRsvdFieldPosition19:   db      0
1298                            <1> MaxPixelClock19:   dd     0
1299                            <1>
1300                            <1>                    dw     018Eh ; 1280x720x24
1301                            <1> ModeAttributes20:  dw     MODE_ATTRIBUTES
1302                            <1> WinAAttributes20:  db     WINA_ATTRIBUTES
1303                            <1> WinBAttributes20:  db     0
1304                            <1> WinGranularity20:  dw     VBE_DISPI_BANK_SIZE_KB
1305                            <1> WinSize20:         dw     VBE_DISPI_BANK_SIZE_KB
1306                            <1> WinASegment20:            dw      VGAMEM_GRAPH
1307                            <1> WinBSegment20:            dw      0000h
1308                            <1> WinFuncPtr20:             dd      0
1309                            <1> BytesPerScanLine20:       dw      3840
1310                            <1> XResolution20:            dw      1280
1311                            <1> YResolution20:            dw      720
1312                            <1> XCharSize20:              db      8
1313                            <1> YCharSize20:              db      16
1314                            <1> NumberOfPlanes20:  db     1
1315                            <1> BitsPerPixel20:           db      24
1316                            <1> NumberOfBanks20:   db     43
1317                            <1> MemoryModel20:            db      VBE_MEMORYMODEL_DIRECT_COLOR
1318                            <1> BankSize20:        db     0
1319                            <1> NumberOfImagePages20:     db      5
1320                            <1> Reserved_page20:   db     0
1321                            <1> RedMaskSize20:            db      8
1322                            <1> RedFieldPosition20:       db      16
1323                            <1> GreenMaskSize20:   db     8
1324                            <1> GreenFieldPosition20:     db      8
1325                            <1> BlueMaskSize20:           db      8
1326                            <1> BlueFieldPosition20:      db      0
1327                            <1> RsvdMaskSize20:           db      0
1328                            <1> RsvdFieldPosition20:      db      0
1329                            <1> DirectColorModeInfo20:    db      0
1330                            <1> PhysBasePtr20:            dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS
1331                            <1> OffScreenMemOffset20:     dd      0
1332                            <1> OffScreenMemSize20:       dw      0
1333                            <1> LinBytesPerScanLine20:    dw      3840
1334                            <1> BnkNumberOfPages20:       db      0
1335                            <1> LinNumberOfPages20:       db      0
1336                            <1> LinRedMaskSize20:  db     8
1337                            <1> LinRedFieldPosition20:    db      16
1338                            <1> LinGreenMaskSize20:       db      8
1339                            <1> LinGreenFieldPosition20:db 8
1340                            <1> LinBlueMaskSize20: db     8
1341                            <1> LinBlueFieldPosition20:   db      0
```

```
                        <1> LinRsvdMaskSize20: db     0
                        <1> LinRsvdFieldPosition20:   db     0
                        <1> MaxPixelClock20:  dd     0
                        <1>
                        <1>                   dw     018Fh ; 1280x720x32
                        <1> ModeAttributes21: dw     MODE_ATTRIBUTES
                        <1> WinAAttributes21: db     WINA_ATTRIBUTES
                        <1> WinBAttributes21: db     0
                        <1> WinGranularity21: dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize21:        dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment21:            dw     VGAMEM_GRAPH
                        <1> WinBSegment21:            dw     0000h
                        <1> WinFuncPtr21:             dd     0
                        <1> BytesPerScanLine21:       dw     5120
                        <1> XResolution21:            dw     1280
                        <1> YResolution21:            dw     720
                        <1> XCharSize21:              db     8
                        <1> YCharSize21:              db     16
                        <1> NumberOfPlanes21: db     1
                        <1> BitsPerPixel21:           db     32
                        <1> NumberOfBanks21:  db     57
                        <1> MemoryModel21:            db     VBE_MEMORYMODEL_DIRECT_COLOR
                        <1> BankSize21:       db     0
                        <1> NumberOfImagePages21:     db     3
                        <1> Reserved_page21:  db     0
                        <1> RedMaskSize21:            db     8
                        <1> RedFieldPosition21:       db     16
                        <1> GreenMaskSize21:  db     8
                        <1> GreenFieldPosition21:     db     8
                        <1> BlueMaskSize21:           db     8
                        <1> BlueFieldPosition21:      db     0
                        <1> RsvdMaskSize21:           db     8
                        <1> RsvdFieldPosition21:      db     24
                        <1> DirectColorModeInfo21:    db     VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
                        <1> PhysBasePtr21:            dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
                        <1> OffScreenMemOffset21:     dd     0
                        <1> OffScreenMemSize21:       dw     0
                        <1> LinBytesPerScanLine21:    dw     5120
                        <1> BnkNumberOfPages21:       db     0
                        <1> LinNumberOfPages21:       db     0
                        <1> LinRedMaskSize21: db     8
                        <1> LinRedFieldPosition21:    db     16
                        <1> LinGreenMaskSize21:       db     8
                        <1> LinGreenFieldPosition21:db 8
                        <1> LinBlueMaskSize21: db     8
                        <1> LinBlueFieldPosition21:   db     0
                        <1> LinRsvdMaskSize21: db     8
                        <1> LinRsvdFieldPosition21:   db     24
                        <1> MaxPixelClock21:  dd     0
                        <1>
                        <1>                   dw     0190h ; 1920x1080x16
                        <1> ModeAttributes22: dw     MODE_ATTRIBUTES
                        <1> WinAAttributes22: db     WINA_ATTRIBUTES
                        <1> WinBAttributes22: db     0
                        <1> WinGranularity22: dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize22:        dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment22:            dw     VGAMEM_GRAPH
                        <1> WinBSegment22:            dw     0000h
                        <1> WinFuncPtr22:             dd     0
                        <1> BytesPerScanLine22:       dw     3840
                        <1> XResolution22:            dw     1920
                        <1> YResolution22:            dw     1080
                        <1> XCharSize22:              db     8
                        <1> YCharSize22:              db     16
                        <1> NumberOfPlanes22: db     1
                        <1> BitsPerPixel22:           db     16
                        <1> NumberOfBanks22:  db     64
                        <1> MemoryModel22:            db     VBE_MEMORYMODEL_DIRECT_COLOR,
                        <1> BankSize22:       db     0
                        <1> NumberOfImagePages22:     db     3
                        <1> Reserved_page22:  db     0
                        <1> RedMaskSize22:            db     5
                        <1> RedFieldPosition22:       db     11
                        <1> GreenMaskSize22:  db     6
                        <1> GreenFieldPosition22:     db     5
                        <1> BlueMaskSize22:           db     5
                        <1> BlueFieldPosition22:      db     0
                        <1> RsvdMaskSize22:           db     0
                        <1> RsvdFieldPosition22:      db     0
                        <1> DirectColorModeInfo22:    db     0
                        <1> PhysBasePtr22:            dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
                        <1> OffScreenMemOffset22:     dd     0
                        <1> OffScreenMemSize22:       dw     0
                        <1> LinBytesPerScanLine22:    dw     3840
                        <1> BnkNumberOfPages22:       db     0
                        <1> LinNumberOfPages22:       db     0
                        <1> LinRedMaskSize22: db     5
                        <1> LinRedFieldPosition22:    db     11
                        <1> LinGreenMaskSize22:       db     6
                        <1> LinGreenFieldPosition22:db 5
                        <1> LinBlueMaskSize22: db     5
                        <1> LinBlueFieldPosition22:   db     0
                        <1> LinRsvdMaskSize22: db     0
                        <1> LinRsvdFieldPosition22:   db     0
                        <1> MaxPixelClock22:  dd     0
                        <1>
                        <1>                   dw     0191h ; 1920x1080x24
                        <1> ModeAttributes23: dw     MODE_ATTRIBUTES
                        <1> WinAAttributes23: db     WINA_ATTRIBUTES
                        <1> WinBAttributes23: db     0
                        <1> WinGranularity23: dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize23:        dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment23:            dw     VGAMEM_GRAPH
                        <1> WinBSegment23:            dw     0000h
                        <1> WinFuncPtr23:             dd     0
                        <1> BytesPerScanLine23:       dw     5760
                        <1> XResolution23:            dw     1920
                        <1> YResolution23:            dw     1080
                        <1> XCharSize23:              db     8
                        <1> YCharSize23:              db     16
                        <1> NumberOfPlanes23: db     1
                        <1> BitsPerPixel23:           db     24
                        <1> NumberOfBanks23:  db     95
                        <1> MemoryModel23:            db     VBE_MEMORYMODEL_DIRECT_COLOR
                        <1> BankSize23:       db     0
                        <1> NumberOfImagePages23:     db     1
                        <1> Reserved_page23:  db     0
                        <1> RedMaskSize23:            db     8
                        <1> RedFieldPosition23:       db     16
                        <1> GreenMaskSize23:  db     8
                        <1> GreenFieldPosition23:     db     8
                        <1> BlueMaskSize23:           db     8
                        <1> BlueFieldPosition23:      db     0
                        <1> RsvdMaskSize23:           db     0
```

```
1466                            <1> RsvdFieldPosition23:      db      0
1467                            <1> DirectColorModeInfo23:    db      0
1468                            <1> PhysBasePtr23:            dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS
1469                            <1> OffScreenMemOffset23:     dd      0
1470                            <1> OffScreenMemSize23:       dw      0
1471                            <1> LinBytesPerScanLine23:    dw      5760
1472                            <1> BnkNumberOfPages23:       db      0
1473                            <1> LinNumberOfPages23:       db      0
1474                            <1> LinRedMaskSize23:  db     8
1475                            <1> LinRedFieldPosition23:    db      16
1476                            <1> LinGreenMaskSize23:       db      8
1477                            <1> LinGreenFieldPosition23:db 8
1478                            <1> LinBlueMaskSize23: db     8
1479                            <1> LinBlueFieldPosition23:   db      0
1480                            <1> LinRsvdMaskSize23: db     0
1481                            <1> LinRsvdFieldPosition23:   db      0
1482                            <1> MaxPixelClock23:   dd     0
1483                            <1>
1484                            <1>                     dw      0192h ; 1920x1080x32
1485                            <1> ModeAttributes24:  dw      MODE_ATTRIBUTES
1486                            <1> WinAAttributes24:  db      WINA_ATTRIBUTES
1487                            <1> WinBAttributes24:  db      0
1488                            <1> WinGranularity24:  dw      VBE_DISPI_BANK_SIZE_KB
1489                            <1> WinSize24:         dw      VBE_DISPI_BANK_SIZE_KB
1490                            <1> WinASegment24:             dw      VGAMEM_GRAPH
1491                            <1> WinBSegment24:             dw      0000h
1492                            <1> WinFuncPtr24:              dd      0
1493                            <1> BytesPerScanLine24:        dw      7680
1494                            <1> XResolution24:             dw      1920
1495                            <1> YResolution24:             dw      1080
1496                            <1> XCharSize24:               db      8
1497                            <1> YCharSize24:               db      16
1498                            <1> NumberOfPlanes24:  db      1
1499                            <1> BitsPerPixel24:            db      32
1500                            <1> NumberOfBanks24:   db      127
1501                            <1> MemoryModel24:             db      VBE_MEMORYMODEL_DIRECT_COLOR
1502                            <1> BankSize24:        db      0
1503                            <1> NumberOfImagePages24:      db      1
1504                            <1> Reserved_page24:   db      0
1505                            <1> RedMaskSize24:             db      8
1506                            <1> RedFieldPosition24:        db      16
1507                            <1> GreenMaskSize24:   db      8
1508                            <1> GreenFieldPosition24:      db      8
1509                            <1> BlueMaskSize24:            db      8
1510                            <1> BlueFieldPosition24:       db      0
1511                            <1> RsvdMaskSize24:            db      8
1512                            <1> RsvdFieldPosition24:       db      24
1513                            <1> DirectColorModeInfo24:     db      VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
1514                            <1> PhysBasePtr24:             dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS
1515                            <1> OffScreenMemOffset24:      dd      0
1516                            <1> OffScreenMemSize24:        dw      0
1517                            <1> LinBytesPerScanLine24:     dw      7680
1518                            <1> BnkNumberOfPages24:        db      0
1519                            <1> LinNumberOfPages24:        db      0
1520                            <1> LinRedMaskSize24:  db      8
1521                            <1> LinRedFieldPosition24:     db      16
1522                            <1> LinGreenMaskSize24:        db      8
1523                            <1> LinGreenFieldPosition24:db 8
1524                            <1> LinBlueMaskSize24: db      8
1525                            <1> LinBlueFieldPosition24:    db      0
1526                            <1> LinRsvdMaskSize24: db      8
1527                            <1> LinRsvdFieldPosition24:    db      24
1528                            <1> MaxPixelClock24:   dd      0
1529                            <1>
1530                            <1> VBE_VESA_MODE_END_OF_LIST: dw      0
1531                            <1>
1532                            <1> %endif
1533                            <1>
1534                            <1> ; 24/11/2020
1535                            <1>
1536                            <1> direct_color_fields:
1537                            <1>     ; 24/11/2020
1538                            <1>
1539                            <1> ; (vbetables-gen.c)
1540                            <1> ; // Direct Color fields
1541                            <1> ; (required for direct/6 and YUV/7 memory models)
1542                            <1> ; switch(pm->depth) {
1543                            <1>
1544                            <1> ;case 8:
1545 00006C6A 00                <1> r_size_8:  db 0
1546 00006C6B 00                <1> r_pos_8:   db 0
1547 00006C6C 00                <1> g_size_8:  db 0
1548 00006C6D 00                <1> g_pos_8:   db 0
1549 00006C6E 00                <1> b_size_8:  db 0
1550 00006C6F 00                <1> b_pos_8:   db 0
1551 00006C70 00                <1> a_size_8:  db 0
1552 00006C71 00                <1> a_pos_8:   db 0
1553                            <1>
1554                            <1> ;case 16:
1555 00006C72 05                <1> r_size_16:  db 5
1556 00006C73 0B                <1> r_pos_16:   db 11
1557 00006C74 06                <1> g_size_16:  db 6
1558 00006C75 05                <1> g_pos_16:   db 5
1559 00006C76 05                <1> b_size_16:  db 5
1560 00006C77 00                <1> b_pos_16:   db 0
1561 00006C78 00                <1> a_size_16:  db 0
1562 00006C79 00                <1> a_pos_16:   db 0
1563                            <1>
1564                            <1> ;case 24:
1565 00006C7A 08                <1> r_size_24:  db 8
1566 00006C7B 10                <1> r_pos_24:   db 16
1567 00006C7C 08                <1> g_size_24:  db 8
1568 00006C7D 08                <1> g_pos_24:   db 8
1569 00006C7E 08                <1> b_size_24:  db 8
1570 00006C7F 00                <1> b_pos_24:   db 0
1571 00006C80 00                <1> a_size_24:  db 0
1572 00006C81 00                <1> a_pos_24:   db 0
1573                            <1>
1574                            <1> ;case 32:
1575 00006C82 08                <1> r_size_32:  db 8
1576 00006C83 10                <1> r_pos_32:   db 16
1577 00006C84 08                <1> g_size_32:  db 8
1578 00006C85 08                <1> g_pos_32:   db 8
1579 00006C86 08                <1> b_size_32:  db 8
1580 00006C87 00                <1> b_pos_32:   db 0
1581 00006C88 08                <1> a_size_32:  db 8
1582 00006C89 18                <1> a_pos_32:   db 24
3425                                ;%include 'diskdata.s'    ; DISK (BIOS) DATA (initialized)
3426                                ;;;
3427
3428                                Align 2
3429
3430                                %include 'sysdefs.s' ; 24/01/2015
   1                            <1> ; ****************************************************************************
```

```
                        <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.5) - SYSTEM DEFINITIONS : sysdefs.s
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Last Update: 23/07/2022  (Previous: 31/12/2017)
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Beginning: 24/01/2016
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Assembler: NASM version 2.11 (trdos386.s)
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
                        <1> ; sysdefs.inc (14/11/2015)
                        <1> ; ****************************************************************************
                        <1>
                        <1> ; Retro UNIX 386 v1 Kernel - SYSDEFS.INC
                        <1> ; Last Modification: 14/11/2015
                        <1> ;
                        <1> ; ///////// RETRO UNIX 386 V1 SYSTEM DEFINITIONS ///////////////
                        <1> ; (Modified from
                        <1> ;   Retro UNIX 8086 v1 system definitions in 'UNIX.ASM', 01/09/2014)
                        <1> ; ((UNIX.ASM (RETRO UNIX 8086 V1 Kernel), 11/03/2013 - 01/09/2014))
                        <1> ;   UNIX.ASM (MASM 6.11) --> SYSDEFS.INC (NASM 2.11)
                        <1> ; ----------------------------------------------------------------------------
                        <1> ;
                        <1> ; Derived from UNIX Operating System (v1.0 for PDP-11)
                        <1> ; (Original) Source Code by Ken Thompson (1971-1972)
                        <1> ; <Bell Laboratories (17/3/1972)>
                        <1> ; <Preliminary Release of UNIX Implementation Document>
                        <1> ;
                        <1> ; ****************************************************************************
                        <1>
                        <1> nproc       equ     16  ; number of processes
                        <1> ;nfiles equ 50
                        <1> nttyequ     8   ; 8+1 -> 8 (10/05/2013)
                        <1> nbufequ     4   ; 6 ;; 21/08/2015 - 'namei' buffer problem when nbuf > 4
                        <1>             ; NOTE: if fd0 super block buffer addres is beyond of the 1st
                        <1>             ; 32K, DMA r/w routine or someting else causes a jump to
                        <1>             ; kernel panic routine (in 'alloc' routine, in u5.s)
                        <1>             ; because of invalid buffer content (r/w error).
                        <1>             ; When all buffers are set before the end of the 1st 32k,
                        <1>             ; there is no problem!? (14/11/2015)
                        <1>
                        <1> ;csgmnt     equ     2000h   ; 26/05/2013 (segment of process 1)
                        <1> ;core       equ     0           ; 19/04/2013
                        <1> ;ecore      equ     32768 - 64  ; 04/06/2013 (24/05/2013)
                        <1>     ; (if total size of argument list and arguments is 128 bytes)
                        <1>     ; maximum executable file size = 32768-(64+40+128-6) = 32530 bytes
                        <1>     ; maximum stack size = 40 bytes (+6 bytes for 'IRET' at 32570)
                        <1>     ; initial value of user's stack pointer = 32768-64-128-2 = 32574
                        <1>     ;       (sp=32768-args_space-2 at the beginning of execution)
                        <1>     ; argument list offset = 32768-64-128 = 32576 (if it is 128 bytes)
                        <1>     ; 'u' structure offset (for the '/core' dump file) = 32704
                        <1>     ; '/core' dump file size = 32768 bytes
                        <1>
                        <1> ; 08/03/2014
                        <1> ;sdsegmnt equ       6C0h  ; 256*16 bytes (swap data segment size for 16 processes)
                        <1> ; 19/04/2013 Retro UNIX 8086 v1 feaure only !
                        <1> ;;sdsegmnt equ     740h  ; swap data segment (for user structures and registers)
                        <1>
                        <1> ; 30/08/2013
                        <1> time_count equ 4 ; 10 --> 4 01/02/2014
                        <1>
                        <1> ; 05/02/2014
                        <1> ; process status
                        <1> ;SFREE      equ 0
                        <1> ;SRUN       equ 1
                        <1> ;SWAIT      equ 2
                        <1> ;SZOMB      equ 3
                        <1> ;SSLEEP     equ 4 ; Retro UNIX 8086 V1 extension (for sleep and wakeup)
                        <1>
                        <1> ; 09/03/2015
                        <1> userdata equ 80000h ; user structure data address for current user ; temporary
                        <1> swap_queue equ 90000h - 2000h ; swap queue address ; temporary
                        <1> swap_alloc_table equ 0D0000h  ;  swap allocation table address ; temporary
                        <1>
                        <1> ; 17/09/2015
                        <1> ESPACE equ 48 ; [u.usp] (at 'sysent') - [u.sp] value for error return
                        <1>
                        <1> ; 31/12/2017
                        <1> ; 19/02/2017
                        <1> ; 15/10/2016
                        <1> ; 20/05/2016
                        <1> ; 19/05/2016
                        <1> ; 18/05/2016
                        <1> ; 29/04/2016
                        <1> ; TRDOS 386 (TRDOS v2.0) system calls - temporary List
                        <1> ; 14/07/2013 - 21/09/2015 (Retro UNIX 8086 & 386 system calls)
                        <1> _ver        equ 0 ; Get TRDOS version (v2.0)
                        <1> _exit       equ 1
                        <1> _fork       equ 2
                        <1> _read       equ 3
                        <1> _write      equ 4
                        <1> _open       equ 5
                        <1> _close      equ 6
                        <1> _wait       equ 7
                        <1> _creat      equ 8
                        <1> _rename     equ 9  ; TRDOS 386, Rename File (31/12/2017)
                        <1> _delete     equ 10 ; TRDOS 386, Delete File (29/12/2017)
                        <1> _exec       equ 11
                        <1> _chdir      equ 12
                        <1> _time       equ 13 ; TRDOS 386, Get Sys Date&Time (30/12/2017)
                        <1> _mkdir      equ 14
                        <1> _chmod      equ 15 ; TRDOS 386, Change Attributes (30/12/2017)
                        <1> _rmdir      equ 16 ; TRDOS 386, Remove Directory (29/12/2017)
                        <1> _break      equ 17
                        <1> _drive      equ 18 ; TRDOS 386, Get/Set Current Drv (30/12/2017)
                        <1> _seek       equ 19
                        <1> _tell       equ 20
                        <1> _memequ 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> _envequ 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> _direqu 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> _msgequ 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)
```

```
126                             <1> _rele       equ 39 ; TRDOS 386 (19/05/2016)
127                             <1> _fffequ 40 ; Find First File - TRDOS 386 (15/10/2016)
128                             <1> _fnfequ 41 ; Find Next File - TRDOS 386 (15/10/2016)
129                             <1> _alloc      equ 42 ; Allocate memory - TRDOS 386 (19/02/2017)
130                             <1>             ; TRDOS 386 (19/02/2017) DMA buff fuctions
131                             <1> _dalloc equ 43 ; Deallocate mem - TRDOS 386 (19/02/2017)
132                             <1> _calbac equ 44 ; Set IRQ callback - TRDOS 386 (20/02/2017)
133                             <1> _dmaequ 45 ; DMA service - TRDOS 386 (20/08/2017)
134                             <1>
135                             <1> %macro sys 1-4
136                             <1>     ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
137                             <1>     ; 03/09/2015
138                             <1>     ; 13/04/2015
139                             <1>     ; Retro UNIX 386 v1 system call.
140                             <1>     %if %0 >= 2
141                             <1>         mov ebx, %2
142                             <1>         %if %0 >= 3
143                             <1>             mov ecx, %3
144                             <1>             %if %0 = 4
145                             <1>                mov edx, %4
146                             <1>             %endif
147                             <1>         %endif
148                             <1>     %endif
149                             <1>     mov eax, %1
150                             <1>     ;int 30h
151                             <1>     int 40h ; TRDOS 386 (TRDOS v2.0)
152                             <1> %endmacro
153                             <1>
154                             <1> ; TRDOS 386 system calls, interrupt number
155                             <1> ; 25/12/2016
156                             <1> SYSCALL_INT_NUM   equ '40' ; '40h'
157                             <1>
158                             <1> ; 13/05/2015 - ERROR CODES
159                             <1> ERR_FILE_NOT_OPEN  equ 10 ; 'file not open !' error
160                             <1> ERR_FILE_ACCESS    equ 11 ; 'permission denied !' error
161                             <1> ; 14/05/2015
162                             <1> ERR_DIR_ACCESS     equ 11 ; 'permission denied !' error
163                             <1> ERR_FILE_NOT_FOUND equ 12 ; 'file not found !' error
164                             <1> ERR_TOO_MANY_FILES equ 13 ; 'too many open files !' error
165                             <1> ERR_DIR_EXISTS     equ 14 ; 'directory already exists !' error
166                             <1> ; 16/05/2015
167                             <1> ERR_DRV_NOT_RDY    equ 15 ; 'drive not ready !' error
168                             <1> ; 18/05/2015
169                             <1> ERR_DEV_NOT_RDY    equ 15 ; 'device not ready !' error
170                             <1> ERR_DEV_ACCESS     equ 11 ; 'permission denied !' error
171                             <1> ERR_DEV_NOT_OPEN   equ 10 ; 'device not open !' error
172                             <1> ; 07/06/2015
173                             <1> ERR_FILE_EOF          equ 16 ; 'end of file !' error
174                             <1> ERR_DEV_VOL_SIZE   equ 16 ; 'out of volume !' error
175                             <1> ; 09/06/2015
176                             <1> ERR_DRV_READ          equ 17 ; 'disk read error !'
177                             <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
```

```
                        <1> MAX_MSG_LEN    equ 255 ; max. msg length for 'sysmsg'
                        <1> ;
                        <1> ; 06/10/2016
                        <1> ;OPENFILES     equ 10  ; max. number of open files (system)
                        <1> ; 23/07/2022
                        <1> OPENFILES      equ 32  ; max. number of open files (system)
                        <1> ; 07/10/2016
                        <1> ;NUMOFDEVICES         equ 20  ; max. num of available devices (sys)
                        <1>
                            %include 'trdosk0.s' ; 04/01/2016
                        <1> ; ****************************************************************************
                        <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - DEFINITIONS : trdosk0.s
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Last Update: 29/02/2016
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Beginning: 04/01/2016
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Assembler: NASM version 2.11 (trdos386.s)
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
                        <1> ; TRDOS2.ASM (09/11/2011)
                        <1> ; ****************************************************************************
                        <1> ; TRDOS2.ASM (c) 2004-2011 Erdogan TAN [ 17/01/2004 ] Last Update: 09/11/2011
                        <1> ;
                        <1> ; Masterboot / Partition Table at Beginning+1BEh
                        <1> ptBootable       equ 0
                        <1> ptBeginHead      equ 1
                        <1> ptBeginSector    equ 2
                        <1> ptBeginCylinder  equ 3
                        <1> ptFileSystemID   equ 4
                        <1> ptEndHead        equ 5
                        <1> ptEndSector      equ 6
                        <1> ptEndCylinder    equ 7
                        <1> ptStartSector    equ 8
                        <1> ptSectors        equ 12
                        <1>
                        <1> ; Boot Sector Parameters at 7C00h
                        <1> DataArea1     equ -4
                        <1> DataArea2     equ -2
                        <1> BootStart     equ 0h
                        <1> OemName       equ 03h
                        <1> BytesPerSec   equ 0Bh
                        <1> SecPerClust   equ 0Dh
                        <1> ResSectors    equ 0Eh
                        <1> FATs          equ 10h
                        <1> RootDirEnts   equ 11h
                        <1> Sectors       equ 13h
                        <1> Media         equ 15h
                        <1> FATSecs       equ 16h
                        <1> SecPerTrack   equ 18h
                        <1> Heads         equ 1Ah
                        <1> Hidden1       equ 1Ch
                        <1> Hidden2       equ 1Eh
                        <1> HugeSec1      equ 20h
                        <1> HugeSec2      equ 22h
                        <1> DriveNumber   equ 24h
                        <1> Reserved1     equ 25h
                        <1> bootsignature equ 26h
                        <1> VolumeID      equ 27h
                        <1> VolumeLabel   equ 2Bh
                        <1> FileSysType   equ 36h
                        <1> Reserved2     equ 3Eh                           ; Starting cluster of P2000
                        <1>
                        <1> ; FAT32 BPB Structure
                        <1> FAT32_FAT_Size equ 36
                        <1> FAT32_RootFClust equ 44
                        <1> FAT32_FSInfoSec equ 48
                        <1> FAT32_DrvNum equ 64
                        <1> FAT32_BootSig equ 66
                        <1> FAT32_VolID equ 67
                        <1> FAT32_VolLab equ 71
                        <1> FAT32_FilSysType equ 82
                        <1>
                        <1> ; BIOS Disk Parameters
                        <1> DPDiskNumber  equ 0h
                        <1> DPDType       equ 1h
                        <1> DPReturn      equ 2h
                        <1> DPHeads       equ 3h
                        <1> DPCylinders   equ 4h
                        <1> DPSecPerTrack equ 6h
                        <1> DPDisks       equ 7h
                        <1> DPTableOff    equ 8h
                        <1> DPTableSeg    equ 0Ah
                        <1> DPNumOfSecs   equ 0Ch
                        <1>
                        <1> ; BIOS INT 13h Extensions (LBA extensions)
                        <1> ; Just After DP Data (DPDiskNumber+)
                        <1> DAP_PacketSize equ 10h  ; If extensions present, this byte will be >=10h
                        <1> DAP_Reserved1 equ 11h   ; Reserved Byte
                        <1> DAP_NumOfBlocks equ 12h ; Value of this byte must be 0 to 127
                        <1> DAP_Reserved2 equ 13h   ; Reserved Byte
                        <1> DAP_Destination equ 14h ; Address of Transfer Buffer as SEGMENT:OFFSET
                        <1> DAP_LBA_Address equ 18h ; LBA=(C1*H0+H1)*S0+S1-1
                        <1>                         ; C1= Selected Cylinder Number
                        <1>                         ; H0= Number Of Heads (Maximum Head Number + 1)
                        <1>                         ; H1= Selected Head Number
                        <1>                         ; S0= Maximum Sector Number
                        <1>                         ; S1= Selected Sector Number
                        <1>                         ; QUAD WORD
                        <1> ; DAP_Flat_Destination equ 20h ; 64 bit address, if value in 4h is FFFF:FFFFh
                        <1>                             ; QUAD WORD (Also, value in 0h must be 18h)
                        <1>                             ; TR-DOS will not use 64 bit Flat Address
                        <1>
                        <1> ; INT 13h Function 48h "Get Enhanced Disk Drive Parameters"
                        <1> ; Just After DP Data (DPDiskNumber+)
                        <1> GetDParams_48h equ 20h ; Word. Data Length, must be 26 (1Ah) for short data.
                        <1> GDP_48h_InfoFlag equ 22h ; Word
                        <1> ; Bit 1 = 1 -> The geometry returned in bytes 4-15 is valid.
                        <1> GDP_48h_NumOfPCyls equ 24h ; Double Word. Number physical cylinders.
                        <1> GDP_48h_NumOfPHeads equ 28h ; Double Word. Number of physical heads.
                        <1> GDP_48h_NumOfPSpT equ 2Ch ; Double word. Num of physical sectors per track.
                        <1> GDP_48h_LBA_Sectors equ 30h ; 8 bytes. Number of physical/LBA sectors.
                        <1> GDP_48h_BytesPerSec equ 38h ; Word. Number of bytes in a sector.
                        <1>
                        <1> ; TR-DOS Standalone Program Extensions to the DiskParams Block
                        <1> ; Just After DP Data (DPDiskNumber+)
                        <1> TRDP_CurrentSector equ 3Ah  ; DX:AX (LBA)
                        <1> TRDP_SectorCount equ 3Eh    ; CX (or Counter)
                        <1>
                        <1>
                        <1> ; DOS Logical Disks
                        <1> LD_Name equ 0
                        <1> LD_DiskType equ 1
                        <1> LD_PhyDrvNo equ 2
```

```
                        <1> LD_FATType equ 3
                        <1> LD_FSType equ 4
                        <1> LD_LBAYes equ 5
                        <1> LD_BPB equ 6
                        <1> LD_FATBegin equ 96
                        <1> LD_ROOTBegin equ 100
                        <1> LD_DATABegin equ 104
                        <1> LD_StartSector equ 108
                        <1> LD_TotalSectors equ 112
                        <1> LD_FreeSectors equ 116
                        <1> LD_Clusters equ 120
                        <1> LD_PartitionEntry equ 124
                        <1> LD_DParamEntry equ 125
                        <1> LD_MediaChanged equ 126
                        <1> LD_CDirLevel equ 127
                        <1> LD_CurrentDirectory equ 128
                        <1>
                        <1> ; Singlix FS Extensions to DOS Logical Disks
                        <1> ; 03/01/2010 (LD_BPB compatibility for CHS r/w)
                        <1>
                        <1> LD_FS_Name equ 0
                        <1> LD_FS_DiskType equ 1
                        <1> LD_FS_PhyDrvNo equ 2
                        <1> LD_FS_FATType equ 3
                        <1> LD_FS_FSType equ 4
                        <1> LD_FS_LBAYes equ 5
                        <1> LD_FS_BPB equ 6
                        <1> LD_FS_MediaAttrib equ 6
                        <1> LD_FS_VersionMajor equ 7
                        <1> LD_FS_RootDirD equ 8
                        <1> LD_FS_MATLocation equ 12
                        <1> LD_FS_Reserved1 equ 16 ;1 reserved byte
                        <1> LD_FS_BytesPerSec equ 17 ; LD_BPB + 0Bh
                        <1> LD_FS_Reserved2 equ 19 ;2 reserved byte
                        <1> LD_FS_DATLocation equ 20
                        <1> LD_FS_DATSectors equ 24
                        <1> LD_FS_Reserved3 equ 28 ;3 reserved word
                        <1> LD_FS_SecPerTrack equ 30 ; LD_BPB + 18h
                        <1> LD_FS_NumHeads equ 32    ; LD_BPB + 1Ah
                        <1> LD_FS_UnDelDirD equ 34
                        <1> LD_FS_Reserved4 equ 38 ;4 reserved word
                        <1> LD_FS_VolumeSerial equ 40
                        <1> LD_FS_VolumeName equ 44
                        <1> LD_FS_BeginSector equ 108
                        <1> LD_FS_VolumeSize equ 112
                        <1> LD_FS_FreeSectors equ 116
                        <1> LD_FS_FirstFreeSector equ 120
                        <1> LD_FS_PartitionEntry equ 124
                        <1> LD_FS_DParamEntry equ 125
                        <1> LD_FS_MediaChanged equ 126
                        <1> LD_FS_CDirLevel equ 127
                        <1> LD_FS_CDIR_Converted equ 128
                        <1>
                        <1> ; Valid FAT Types
                        <1> FS_FAT12 equ 1
                        <1> FS_FAT16_CHS equ 2
                        <1> FS_FAT32_CHS equ 3
                        <1> FS_FAT16_LBA equ 4
                        <1> FS_FAT32_LBA equ 5
                        <1>
                        <1> ; Cursor Location
                        <1> CCCpointer equ  0450h   ; BIOS data, current cursor column
                        <1> ; FAT Clusters EOC sign
                        <1> FAT12EOC equ 0FFFh
                        <1> FAT16EOC equ 0FFFFh
                        <1> ;FAT32EOC equ 0FFFFFFFh ; It is not direct usable for 8086 code
                        <1> ; BAD Cluster
                        <1> FAT12BADC equ 0FF7h
                        <1> FAT16BADC equ 0FFF7h
                        <1> ;FAT32BADC equ 0FFFFFF7h ; It is not direct usable for 8086 code
                        <1> ; MS-DOS FAT16 FS (Maximum Possible) Last Cluster Number= 0FFF6h
                        <1>
                        <1> ; TRFS
                        <1>
                        <1> bs_FS_JmpBoot equ 0 ; jmp short bsBootCode
                        <1>                     ; db 0EBh, db 3Fh, db 90h
                        <1> bs_FS_Identifier equ 3  ; db 'FS', db 0
                        <1> bs_FS_BytesPerSec equ 6 ; dw 512
                        <1> bs_FS_MediaAttrib equ 8 ; db 3
                        <1> bs_FS_PartitionID equ 9 ; db 0A1h
                        <1> bs_FS_VersionMaj equ 10 ; db 01h
                        <1> bs_FS_VersionMin equ 11 ; db 0
                        <1> bs_FS_BeginSector equ 12   ; dd 0
                        <1> bs_FS_VolumeSize equ 16 ; dd 2880
                        <1> bs_FS_StartupFD equ 20 ; dd 0
                        <1> bs_FS_MATLocation equ 24 ; dd 1
                        <1> bs_FS_RootDirD equ 28 ; dd 8
                        <1> bs_FS_SystemConfFD equ 32 ; dd 0
                        <1> bs_FS_SwapFD equ 36 ; dd 0
                        <1> bs_FS_UnDelDirD equ 40 ; dd 0
                        <1> bs_FS_DriveNumber equ 44 ; db 0
                        <1> bs_FS_LBA_Ready equ 45 ; db 0
                        <1> bs_FS_MagicWord equ 46
                        <1> bs_FS_SecPerTrack equ 46 ; db 0A1h
                        <1> bs_FS_Heads equ 47 ; db 01h
                        <1> bs_FS_OperationSys equ 48 ; db "TR-SINGLIX v1.0b"
                        <1> bs_FS_Terminator equ 64 ; db 0
                        <1> bs_FS_BootCode equ 65
                        <1>
                        <1> FS_MAT_DATLocation equ 12
                        <1> FS_MAT_DATScount equ 16
                        <1> FS_MAT_FreeSectors equ 20
                        <1> FS_MAT_FirstFreeSector equ 24
                        <1> FS_RDT_VolumeSerialNo equ 28
                        <1> FS_RDT_VolumeName equ 64
                        <1>
                        <1> ; FAT12 + FAT16 + FAT32
                        <1> BS_JmpBoot equ 0
                        <1> BS_OEMName equ 3
                        <1> BPB_BytsPerSec equ 11
                        <1> BPB_SecPerClust equ 13
                        <1> BPB_RsvdSecCnt equ 14
                        <1> BPB_NumFATs equ 16
                        <1> BPB_RootEntCnt equ 17
                        <1> BPB_TotalSec16 equ 19
                        <1> BPB_Media equ 21
                        <1> BPB_FATSz16 equ 22
                        <1> BPB_SecPerTrk equ 24
                        <1> BPB_NumHeads equ 26
                        <1> BPB_HiddSec equ 28
                        <1> BPB_TotalSec32 equ 32
                        <1>
                        <1> ; FAT12 and FAT16 only
                        <1> BS_DrvNum equ 36
```

```
                        <1> BS_Reserved1 equ 37
                        <1> BS_BootSig equ 38
                        <1> BS_VolID equ 39
                        <1> BS_VolLab equ 43
                        <1> BS_FilSysType equ 54 ; 8 bytes
                        <1> BS_BootCode equ 62
                        <1>
                        <1> ; FAT32 only
                        <1> BPB_FATSz32 equ 36 ; FAT32, 4 bytes
                        <1> BPB_ExtFlags equ 40 ; FAT32, 2 bytes
                        <1> BPB_FSVer equ 42 ; FAT32, 2 bytes
                        <1> BPB_RootClus equ 44 ; FAT32, 4 bytes
                        <1> BPB_FSInfo equ 48 ; FAT 32, 2 bytes
                        <1> BPB_BkBootSec equ 50 ; FAT32, 2 bytes
                        <1> BPB_Reserved equ 52 ; FAT32, 12 bytes
                        <1> BS_FAT32_DrvNum equ 64 ; FAT32, 1 byte
                        <1> BS_FAT32_Reserved1 equ 65 ; FAT32, 1 byte
                        <1> BS_FAT32_BootSig equ 66 ; FAT32, 1 byte
                        <1> BS_FAT32_VolID equ 67 ; FAT32, 4 bytes
                        <1> BS_FAT32_VolLab equ 71 ; FAT32, 11 bytes
                        <1> BS_FAT32_FilSysType equ 82 ; FAT32, 8 bytes
                        <1> BS_FAT32_BootCode equ 90
                        <1>
                        <1> ; 29/02/2016
                        <1> ;(FAT32 Free Cluster Count & First Free Cluster values)
                        <1> ;[BPB_Reserved] = Free Cluster Count (offset 52)
                        <1> ;[BPB_Reserved+4] = First Free Cluster (offset 56)
                        <1>
                        <1> BS_Validation equ 510
                        <1>
                        <1> ; 15/02/2016
                        <1> ; FILE.ASM - 09/10/2011
                        <1> ; Directory Entry Structure
                        <1> ; 29/10/2009 (According to Microsoft FAT32 File System Specification)
                        <1> DirEntry_Name equ 0
                        <1> DirEntry_Attr equ 11
                        <1> DirEntry_NTRes equ 12
                        <1> DirEntry_CrtTimeTenth equ 13
                        <1> DirEntry_CrtTime equ 14
                        <1> DirEntry_CrtDate equ 16
                        <1> DirEntry_LastAccDate equ 18
                        <1> DirEntry_FstClusHI equ 20
                        <1> DirEntry_WrtTime equ 22
                        <1> DirEntry_WrtDate equ 24
                        <1> DirEntry_FstClusLO equ 26
                        <1> DirEntry_FileSize equ 28
                            %include 'trdosk1.s' ; 04/01/2016
                        <1> ; ****************************************************************************
                        <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.9) - SYS INIT : trdosk1.s
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Last Update: 26/09/2024 (Previous: 25/07/2022)
                        <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>     ; 26/09/2024 (TRDOS v2.0.9)
                        <1>     ; 18/04/2021 (TRDOS 386 v2.0.4)
                        <1>     ; 20/01/2018  (v2.0.1)
                        <1>     ; 23/01/2017  (v2.0.0)
                        <1>     ; 07/05/2016
                        <1>     ; 02/05/2016
                        <1>     ; 24/04/2016
                        <1>     ; 14/04/2016
                        <1>     ; 13/04/2016
                        <1>     ; 30/03/2016
                        <1>     ; 24/01/2016
                        <1>     ; 06/01/2016
                        <1>     ; 04/01/2016 (TRDOS 386 v2.0 - Beginning)
                        <1>
                        <1>     ; 23/01/2017 - reset timer frequency (to 18.2Hz)
 00006C8A B036          <1>     mov     al, 00110110b ; 36h
 00006C8C E643          <1>     out     43h, al
 00006C8E 31C0          <1>     xor     eax, eax  ; sub       al, al ; 0
 00006C90 E640          <1>     out     40h, al ; LB
 00006C92 E640          <1>     out     40h, al ; HB
                        <1>     ;
                        <1>     ; 30/03/2016
                        <1>     ; Clear Logical DOS Disk Description Tables Area
                        <1>     ;xor    eax, eax
 00006C94 BF00010900    <1>     mov     edi, Logical_DOSDisks
 00006C99 B980060000    <1>     mov     ecx, 6656/4 ; 26*256 = 6656 bytes
 00006C9E F3AB          <1>     rep     stosd ; 1664 times 4 bytes
                        <1>
 00006CA0 B83F3A2F00    <1>     mov     eax, '?:/'
 00006CA5 A3[43770100]  <1>     mov     [Current_Dir_Drv], eax
                        <1>
                        <1>     ; Logical DRV INIT (only for hard disks)
 00006CAA E8C3030000    <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>
 00006CAF BE00010900    <1>     mov     esi, Logical_DOSDisks
 00006CB4 B001          <1>     mov     al, 1 ; Initialization sign (invalid_fd_parameter)
 00006CB6 83C67E        <1>     add     esi, LD_MediaChanged ; Media Change Status = 1 (init needed)
 00006CB9 8806          <1>     mov     [esi], al ; A:
                        <1>     ; Temporary - 26/09/2024
                        <1>     ;;mov   dword [esi+LD_Clusters], -1 ; *
                        <1>     ;dec    dword [esi+LD_Clusters]
 00006CBB 81C600010000  <1>     add     esi, 100h
 00006CC1 8806          <1>     mov     [esi], al ; B:
                        <1>     ;;mov   dword [esi+LD_Clusters], -1 ; *
                        <1>     ;dec    dword [esi+LD_Clusters]
                        <1>
                        <1> _current_drive_bootdisk:
 00006CC3 8A15[78660000] <1>     mov     dl, [boot_drv] ; physical drive number
 00006CC9 80FAFF        <1>     cmp     dl, 0FFh
 00006CCC 740A          <1>     je      short _last_dos_diskno_check
                        <1> _boot_drive_check:
 00006CCE 80FA80        <1>     cmp     dl, 80h
 00006CD1 7218          <1>     jb      short _current_drive_a
 00006CD3 80EA7E        <1>     sub     dl, 7Eh ; C = 2 , D = 3
 00006CD6 EB13          <1>     jmp     short _current_drive_a
                        <1>
                        <1> _last_dos_diskno_check:
```

```
 78 00006CD8 8A15[5C2E0100]      <1>     mov     dl, [Last_DOS_DiskNo]
 79 00006CDE 80FA02              <1>     cmp     dl, 2
 80 00006CE1 7706                <1>     ja      short _current_drive_c
 81 00006CE3 7406                <1>     je      short _current_drive_a
 82 00006CE5 30D2                <1>     xor     dl, dl ; A:
 83 00006CE7 EB02                <1>     jmp     short _current_drive_a
 84                              <1>
 85                              <1> _current_drive_c:
 86 00006CE9 B202                <1>     mov     dl, 2 ; C:
 87                              <1>
 88                              <1> _current_drive_a:
 89 00006CEB 8815[79660000]      <1>     mov     [drv], dl
 90 00006CF1 BE[5E2E0100]        <1>         mov     esi, msg_CRLF_temp
 91 00006CF6 E8AF000000          <1>     call    print_msg
 92                              <1>
 93 00006CFB 8A15[79660000]      <1>     mov     dl, [drv]
 94                              <1> _default_drive_c:
 95 00006D01 E8DA090000          <1>     call    change_current_drive
 96 00006D06 731C                <1>     jnc     short _start_mainprog
 97                              <1>
 98                              <1> _drv_not_ready_error:
 99 00006D08 BE[22310100]        <1>     mov     esi, msgl_drv_not_ready
100 00006D0D E898000000          <1>     call    print_msg
101                              <1>         ;jmp       _end_of_mainprog
102                              <1>
103                              <1>     ; 20/01/2018
104 00006D12 B202                <1>     mov     dl, 2
105 00006D14 3815[79660000]      <1>     cmp     [drv], dl
106 00006D1A 736C                <1>     jnb     short _end_of_mainprog
107 00006D1C 8815[79660000]      <1>     mov     [drv], dl
108 00006D22 EBDD                <1>     jmp     short _default_drive_c
109                              <1>
110                              <1> _start_mainprog:
111                              <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
112                              <1>     ; 18/04/2021 (TRDOS 386 v2.0.4 - Beginning)
113                              <1>     ; 07/01/2017
114                              <1>     ; 07/05/2016
115                              <1>     ; 02/05/2016
116                              <1>     ; 24/04/2016 (TRDOS 386 v2)
117                              <1>     ; Retro UNIX 386 v1, 'sys_init' (u0.s)
118                              <1>     ; 23/06/2015
119                              <1>
120                              <1>     ; 02/05/2016
121                              <1>     ; 24/04/2016
122                              <1>     ;mov    ax, 1
123                              <1>     ; 18/04/2021 - TRDOS 386 v2.0.4
124 00006D24 31C0                <1>     xor     eax, eax
125 00006D26 FEC0                <1>     inc     al  ; eax = 1
126 00006D28 A2[858E0100]        <1>     mov     [u.uno], al
127 00006D2D 66A3[1C8E0100]      <1>     mov     [mpid], ax
128 00006D33 66A3[088D0100]      <1>     mov     [p.pid], ax
129 00006D39 A2[688D0100]        <1>     mov     [p.stat], al
130 00006D3E C605[7C8E0100]04    <1>     mov     byte [u.quant], time_count  ; 07/01/2017
131                              <1>     ;
132 00006D45 A1[80760100]        <1>     mov     eax, [k_page_dir]
133 00006D4A A3[8C8E0100]        <1>     mov     [u.pgdir], eax ; reset
134                              <1>     ;
135 00006D4F E8D2EAFFFF          <1>     call    allocate_page
136                              <1>     ;jc     panic
137                              <1>     ; 25/07/2022
138 00006D54 7305                <1>     jnc     short _start_mainprog_1
139 00006D56 E9AE000000          <1>     jmp     panic
140                              <1>
141                              <1> _start_mainprog_1:
142 00006D5B A3[888E0100]        <1>     mov     [u.upage], eax ; user structure page
143 00006D60 A3[788D0100]        <1>     mov     [p.upage], eax
144 00006D65 E82DEBFFFF          <1>     call    clear_page
145                              <1>     ;
146                              <1>     ; 24/08/2015
147 00006D6A FE0D[288E0100]      <1>     dec     byte [sysflg] ; FFh = ready for system call
148                              <1>                           ; 0 = executing a system call
149                              <1>     ; 13/04/2016
150                              <1>     ; Clear Environment Variables Page/Area
151 00006D70 BF00300900          <1>     mov     edi, Env_Page ; 93000h
152 00006D75 B980000000          <1>     mov     ecx, Env_Page_Size / 4; 512/4  (4096/4)
153 00006D7A 31C0                <1>     xor     eax, eax
154 00006D7C F3AB                <1>     rep     stosd
155                              <1>
156                              <1>     ; 14/04/2016
157 00006D7E E880300000          <1>     call    mainprog_startup_configuration
158                              <1>
159 00006D83 E8870A0000          <1>         call    dos_prompt
160                              <1>
161                              <1> _end_of_mainprog:
162 00006D88 BE[5E2E0100]        <1>         mov     esi, msg_CRLF_temp
163 00006D8D E818000000          <1>     call    print_msg
164 00006D92 BE[642E0100]        <1>     mov     esi, mainprog_Version
165 00006D97 E80E000000          <1>     call    print_msg
166                              <1>     ; 24/01/2016
167 00006D9C 28E4                <1>     sub     ah, ah
168 00006D9E E879A1FFFF          <1>     call    int16h ; call getch
169 00006DA3 E907A6FFFF          <1>     jmp     cpu_reset
170                              <1>
171 00006DA8 EBFE                <1> infinitiveloop: jmp short infinitiveloop
172                              <1>
173                              <1> print_msg:
174                              <1>     ; 13/05/2016
175                              <1>     ; 04/01/2016
176                              <1>     ; 01/07/2015
177                              <1>     ; 13/03/2015 (Retro UNIX 386 v1)
178                              <1>     ; 07/03/2014 (Retro UNIX 8086 v1)
179                              <1>     ; (Modified registers: EAX, EBX, ECX, EDX, ESI, EDI)
180                              <1>     ;
181 00006DAA 8A3D[AE760100]      <1>     mov     bh, [ACTIVE_PAGE] ; 04/01/2016 (ptty)
182                              <1>     ;mov    bl, 07h ; Black background, light gray forecolor
183                              <1>
184 00006DB0 AC                  <1>     lodsb
185                              <1> pmsg1:
186 00006DB1 56                  <1>     push    esi
187                              <1>     ;mov    bh, [ACTIVE_PAGE] ; 04/01/2016 (ptty)
188 00006DB2 B307                <1>     mov     bl, 07h ; Black background, light gray forecolor
189 00006DB4 E822B5FFFF          <1>     call    _write_tty
190 00006DB9 5E                  <1>     pop     esi
191 00006DBA AC                  <1>     lodsb
192 00006DBB 20C0                <1>     and     al, al
193 00006DBD 75F2                <1>     jnz     short pmsg1
194 00006DBF C3                  <1>     retn
195                              <1>
196                              <1> clear_screen:
197                              <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
198                              <1>     ; 06/12/2020
199                              <1>     ; 03/12/2020 (TRDOS 386 v2.0.3)
200                              <1>     ; 13/05/2016
201                              <1>     ; 30/01/2016
```

```
                                <1>     ; 24/01/2016
                                <1>     ; 04/01/2016
00006DC0 0FB61D[AE760100]       <1>     movzx   ebx, byte [ACTIVE_PAGE] ; video page number (0 to 7)
                                <1>     ; 25/07/2022
                                <1>     ;sub    al, al
                                <1>     ; al = 0
00006DC7 8AA3[2F680000]         <1>     mov     ah, [ebx+vmode] ; default = 03h (80x25 text)
00006DCD 80FC04                 <1>     cmp     ah, 4
00006DD0 7205                   <1>     jb      short cls1
00006DD2 80FC07                 <1>     cmp     ah, 7
00006DD5 752B                   <1>     jne     short vga_clear
                                <1> cls1:
                                <1>     ;mov    bh, bl
                                <1>     ;mov    bl, 7
00006DD7 3A25[1E680000]         <1>     cmp     ah, [CRT_MODE] ; current video mode ?
00006DDD 740D                   <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
                                <1>     ;cmp    byte [pmi32], 0
                                <1>     ; 25/07/2022
00006DDF 383D[EC9C0100]         <1>     cmp     [pmi32], bh ; 0
00006DE5 771B                   <1>     ja      short vga_clear
00006DE7 E9F9B4FFFF             <1>     jmp     set_mode_3
                                <1> cls2:
00006DEC 88DF                   <1>     mov     bh, bl ; video page (0 to 7)
00006DEE B307                   <1>     mov     bl, 07h ; attribute to be used on blanked line
                                <1>     ; 25/07/2022
                                <1>     ;sub    al, al ; 0 = entire window
                                <1>     ;xor    cx, cx
                                <1>     ; 25/07/2022
                                <1>     ; al = 0
00006DF0 31C9                   <1>     xor     ecx, ecx
00006DF2 66BA4F18               <1>     mov     dx, 184Fh
00006DF6 E832B2FFFF             <1>     call    _scroll_up ; 24/01/2016
                                <1>     ;
                                <1>     ;mov    bh, [ACTIVE_PAGE] ; video page number (0 to 7)
                                <1>     ;xor    dx, dx
                                <1>     ; 25/07/2022
00006DFB 31D2                   <1>     xor     edx, edx
                                <1>     ;call   _set_cpos ; 24/01/2016
                                <1>     ;;retn
                                <1>     ; 03/12/2020
00006DFD E971B5FFFF             <1>     jmp     _set_cpos ; returns to the caller of this proc
                                <1> ;cls3:
                                <1> ;   retn
                                <1>
                                <1>     ; 06/12/2020
                                <1> vga_clear:
                                <1>     ; 03/12/2020
                                <1>     ; set mode by using _int10h
                                <1>     ; (also clears screen)
                                <1>     ;mov    al, ah
                                <1>     ;sub    ah, ah  ; set current video mode
                                <1>     ; 25/07/2022
00006E02 86C4                   <1>     xchg    ah, al
                                <1>     ; ah = 0
                                <1>     ; al = video mode
                                <1>     ;call   _int10h ; simulates int 10h in TRDOS 386 kernel
                                <1>     ;jmp    short cls3
00006E04 E91BA9FFFF             <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)
00006E09 BE[0F390100]           <1>     mov     esi, panic_msg
00006E0E E897FFFFFF             <1>     call    print_msg
                                <1> key_to_reboot:
                                <1>         ; 24/01/2016
00006E13 28E4                   <1>         sub     ah, ah
00006E15 E802A1FFFF             <1>         call    int16h    ; call getch
                                <1>         ; wait for a character from the current tty
                                <1>     ;
00006E1A B00A                   <1>     mov     al, 0Ah
00006E1C 8A3D[AE760100]         <1>     mov     bh, [ptty] ; [ACTIVE_PAGE]
00006E22 B307                   <1>     mov     bl, 07h ; Black background,
                                <1>                     ; light gray forecolor
00006E24 E8B2B4FFFF             <1>     call    _write_tty
00006E29 E981A5FFFF             <1>     jmp     cpu_reset
                                <1>
                                <1> ctrlbrk:
                                <1>     ; 21/08/2024 (TRDOS 386 Kernel v2.0.9)
                                <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <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>             ;
                                <1>
                                <1>     ; 25/07/2022
00006E2E 52                     <1>     push    edx
00006E2F 31D2                   <1>     xor     edx, edx
                                <1>
                                <1> ; 21/08/2024 - TRDOS 386 v2.0.9
                                <1> %if 0
                                <1>     ;cmp    word [u.intr], 0
                                <1>     cmp     [u.intr], dx ; 0
                                <1>     jna     short cbrk4
                                <1> cbrk0:
                                <1>     ; 12/11/2015
                                <1>     ; 06/12/2013
                                <1>     ;cmp    word [u.quit], 0
                                <1>     cmp     [u.quit], dx ; 0 ; 25/07/2022
                                <1>     jz      short cbrk4
                                <1> %endif
                                <1>     ; 20/09/2013
                                <1>     ;push   ax
00006E31 50                     <1>     push    eax ; 25/07/2022
00006E32 A0[AE760100]           <1>     mov     al, [ptty]
                                <1>
                                <1> ; 21/08/2024 - TRDOS 386 v2.0.9
                                <1> %if 1
00006E37 3A05[6A8E0100]         <1>     cmp     al, [u.ttyn]
00006E3D 7509                   <1>     jne     short cbrk3
                                <1> %else
                                <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>
                                <1>     cmp     al, 8 ; serial port tty nums > 7
                                <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>
                                <1>     ; 25/07/2022
                                <1>     ;push   edx
                                <1>     ;movzx  edx, byte [u.uno]
                                <1>     mov     dl, [u.uno]
                                <1>     cmp     al, [edx+p.ttyc-1] ; console tty (rw)
                                <1>     ;pop    edx
                                <1>     je      short cbrk2
                                <1> cbrk1:
                                <1>     inc     al  ; [u.ttyp] : 1 based tty number
                                <1>     ; 06/12/2013
                                <1>     cmp     al, [u.ttyp]    ; recent open tty (r)
                                <1>     je      short cbrk2
                                <1>         cmp     al, [u.ttyp+1] ; recent open tty (w)
                                <1>     jne     short cbrk3
                                <1> cbrk2:
                                <1>     ;; 06/12/2013
                                <1>     ;mov    ax, [u.quit]
                                <1>     ;and    ax, ax
                                <1>     ;jz     short cbrk3
                                <1> %endif
                                <1>     ;xor    ax, ax ; 0
                                <1>     ;dec    ax
                                <1>     ; 0FFFFh = 'ctrl+brk' keystroke
                                <1>     ; 25/07/2022
                                <1>     ;xor    eax, eax ; 0
                                <1>     ;dec    eax ; -1 ; 0FFFFFFFFh
                                <1>     ;mov    [u.quit], ax
                                <1>     ; 21/08/2024
                                <1>     ; set CTRL+BREAK flag (even if it is not activated)
                                <1>     ; (u.intr is it's activation flag, 0 = disabled))
00006E3F 66C705[828E0100]FF-    <1>     mov     word [u.quit], -1 ; 0FFFFh
00006E47 FF                     <1>
                                <1> cbrk3:
                                <1>     ;pop    ax
00006E48 58                     <1>     pop     eax ; 25/07/2022
                                <1> cbrk4:
                                <1>     ; 25/07/2022
00006E49 5A                     <1>     pop     edx
00006E4A C3                     <1>     retn
                                <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>     ; 18/04/2021 (TRDOS 386 v2.0.3)
                                <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>
                                <1>     ; 18/04/2021
                                <1>     ; INPUT:
                                <1>     ;       none (real time clock)
                                <1>     ; OUTPUT:
                                <1>     ;       eax = unix epoch time value
                                <1>     ;           (seconds since 1/1/1970 00:00:00)
                                <1>
00006E4B E801F5FFFF             <1>     call    get_rtc_time          ; Return Current Time
                                <1>         ;xchg      ch, cl ; 18/04/2021
00006E50 66890D[4A730100]       <1>         mov        [hour], cx    ; BCD, cl = minute, ch = hour
                                <1>         ;xchg      dh, dl ; 18/04/2021
                                <1>     ;mov    [second], dx  ; BCD, dh = second, dl = dse
                                <1>     ; 18/04/2021
00006E57 8835[4E730100]         <1>     mov     [second], dh  ; second
                                <1>     ;
00006E5D E85AF5FFFF             <1>         call       get_rtc_date          ; Return Current Date
                                <1>         ;xchg      ch, cl ; 18/04/2021
00006E62 66890D[44730100]       <1>         mov        [year], cx    ; BCD, cl = year, ch = century
                                <1>         ;xchg      dh, dl ; 18/04/2021
00006E69 668915[46730100]       <1>         mov        [month], dx   ; BCD, dl = day, dh = month
                                <1>     ;
                                <1>     ;mov    al, [hour]    ; Hour
                                <1>         ; 18/04/2021
00006E70 A0[4B730100]           <1>     mov     al, [hour+1]  ; Hour
                                <1>             ; AL <-- BCD number
00006E75 D410                   <1>         db  0D4h, 10h             ; Undocumented inst. AAM
                                <1>                                   ; AH = AL / 10h
                                <1>                                   ; AL = AL MOD 10h
00006E77 D50A                   <1>         aad         ; AX= AH*10+AL
                                <1>     ;mov    [hour], al
                                <1>     ;mov    al, [hour+1]  ; Minute
00006E79 8605[4A730100]         <1>     xchg    al, [hour]    ; [hour] = hour, al = minute
                                <1>             ; AL <-- BCD number
00006E7F D410                   <1>         db  0D4h, 10h             ; Undocumented inst. AAM
                                <1>                                   ; AH = AL / 10h
                                <1>                                   ; AL = AL MOD 10h
00006E81 D50A                   <1>         aad         ; AX= AH*10+AL
00006E83 A2[4C730100]           <1>     mov     [minute], al
00006E88 A0[4E730100]           <1>     mov     al, [second]  ; Second
                                <1>             ; AL <-- BCD number
00006E8D D410                   <1>         db  0D4h, 10h             ; Undocumented inst. AAM
                                <1>                                   ; AH = AL / 10h
                                <1>                                   ; AL = AL MOD 10h
```

```
00006E8F D50A                <1>         aad        ; AX= AH*10+AL
00006E91 A2[4E730100]        <1>     mov     [second], al
00006E96 66A1[44730100]      <1>     mov     ax, [year]    ; Year (century)
                             <1>     ; 18/04/2021
                             <1>     ;push   eax ; puhs ax
                             <1>     ;mov    al, ah ; century ; 18/04/2021
                             <1>             ; AL <-- BCD number
00006E9C D410                <1>         db  0D4h, 10h              ; Undocumented inst. AAM
                             <1>                                    ; AH = AL / 10h
                             <1>                                    ; AL = AL MOD 10h
00006E9E D50A                <1>         aad        ; AX= AH*10+AL
                             <1>     ;mov    ah, 100
                             <1>     ;mul    ah
                             <1>     ;mov    [year], ax
                             <1>     ; 18/04/2021
                             <1>     ; ax = al = year (0 to 99)
00006EA0 668705[44730100]    <1>     xchg    ax, [year]    ; [year+1] = century -> ah
                             <1>     ;pop    eax ; pop ax
00006EA7 88E0                <1>     mov     al, ah  ; century
                             <1>             ; AL <-- BCD number
00006EA9 D410                <1>         db  0D4h, 10h              ; Undocumented inst. AAM
                             <1>                                    ; AH = AL / 10h
                             <1>                                    ; AL = AL MOD 10h
00006EAB D50A                <1>         aad        ; AX= AH*10+AL
                             <1>     ; 18/04/2021
00006EAD B464                <1>     mov     ah, 100       ; 100*(century byte of year)
00006EAF F6E4                <1>     mul     ah
                             <1>     ;
00006EB1 660105[44730100]    <1>     add     [year], ax
                             <1>     ;mov    al, [month]   ; Month
                             <1>     ; 18/04/2021
00006EB8 A0[47730100]        <1>     mov     al, [month+1] ; Month
                             <1>             ; AL <-- BCD number
00006EBD D410                <1>         db  0D4h, 10h              ; Undocumented inst. AAM
                             <1>                                    ; AH = AL / 10h
                             <1>                                    ; AL = AL MOD 10h
00006EBF D50A                <1>         aad        ; AX= AH*10+AL
                             <1>     ;mov    [month], al
                             <1>         ;mov       al, [month+1] ; Day
                             <1>     ; 18/04/2021
00006EC1 8605[46730100]      <1>     xchg    al, [month]   ; [month] = month, al = day
                             <1>             ; AL <-- BCD number
00006EC7 D410                <1>         db  0D4h, 10h              ; Undocumented inst. AAM
                             <1>                                    ; AH = AL / 10h
                             <1>                                    ; AL = AL MOD 10h
00006EC9 D50A                <1>         aad        ; AX= AH*10+AL
00006ECB A2[48730100]        <1>         mov     [day], al
                             <1>
00006ED0 C3                  <1>     retn    ; 30/12/2017
                             <1>
                             <1> epoch:
00006ED1 E875FFFFFF          <1>     call    get_rtc_date_time ; TRDOS 386 - 30/12/2017
                             <1>
                             <1> convert_to_epoch:
                             <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
                             <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
00006ED6 29C0                <1>     sub     eax, eax
00006ED8 66A1[44730100]      <1>     mov     ax, [year]
00006EDE 662DB207            <1>     sub     ax, 1970
00006EE2 BA6D010000          <1>     mov     edx, 365
00006EE7 F7E2                <1>     mul     edx
00006EE9 31DB                <1>     xor     ebx, ebx
00006EEB 8A1D[46730100]      <1>     mov     bl, [month]
00006EF1 FECB                <1>     dec     bl
00006EF3 D0E3                <1>     shl     bl, 1
                             <1>     ;sub    edx, edx
00006EF5 668B93[50730100]    <1>     mov     dx, [EBX+DMonth]
00006EFC 8A1D[48730100]      <1>         mov     bl, [day]
00006F02 FECB                <1>     dec     bl
00006F04 01D0                <1>     add     eax, edx
00006F06 01D8                <1>     add     eax, ebx
                             <1>                     ; EAX = days since 1/1/1970
00006F08 668B15[44730100]    <1>     mov     dx, [year]
00006F0F 6681EAB107          <1>     sub     dx, 1969
                             <1>     ;shr    dx, 1
                             <1>     ;shr    dx, 1
                             <1>     ; 25/07/2022
00006F14 C1EA02              <1>     shr     edx, 2
                             <1>             ; (year-1969)/4
00006F17 01D0                <1>     add     eax, edx
                             <1>                     ; + leap days since 1/1/1970
00006F19 803D[46730100]02    <1>     cmp     byte [month], 2       ; if past february
00006F20 760E                <1>     jna     short cte1
00006F22 668B15[44730100]    <1>     mov     dx, [year]
00006F29 6683E203            <1>     and     dx, 3 ; year mod 4
00006F2D 7501                <1>     jnz     short cte1
                             <1>                     ; and if leap year
                             <1>     ;add    eax, 1  ; add this year's leap day (february 29)
                             <1>     ; 25/07/2022
00006F2F 40                  <1>     inc     eax
                             <1> cte1:                       ; compute seconds since 1/1/1970
00006F30 BA18000000          <1>     mov     edx, 24
00006F35 F7E2                <1>     mul     edx
00006F37 8A15[4A730100]      <1>     mov     dl, [hour]
00006F3D 01D0                <1>     add     eax, edx
                             <1>             ; EAX = hours since 1/1/1970 00:00:00
                             <1>     ;mov    ebx, 60
00006F3F B33C                <1>     mov     bl, 60
00006F41 F7E3                <1>     mul     ebx
00006F43 8A15[4C730100]      <1>     mov     dl, [minute]
00006F49 01D0                <1>     add     eax, edx
                             <1>             ; EAX = minutes since 1/1/1970 00:00:00
                             <1>     ;mov    ebx, 60
00006F4B F7E3                <1>     mul     ebx
00006F4D 8A15[4E730100]      <1>     mov     dl, [second]
00006F53 01D0                <1>     add     eax, edx
                             <1>             ; EAX -> seconds since 1/1/1970 00:00:00
00006F55 C3                  <1>     retn
                             <1>
                             <1> ;set_date_time:
                             <1> convert_from_epoch:
                             <1>     ; 25/07/2022 (v2.0.5)
                             <1>     ; 18/04/2021 (v2.0.4)
                             <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>     ;
00006F56 31D2                   <1>     xor     edx, edx
                                <1>     ;mov    ecx, 60
                                <1>     ; 25/07/2022
00006F58 29C9                   <1>     sub     ecx, ecx
00006F5A B13C                   <1>     mov     cl, 60
00006F5C F7F1                   <1>     div     ecx
                                <1>     ;mov    [imin], eax    ; whole minutes
                                <1>                             ; since 1/1/1970
00006F5E 668915[4E730100]       <1>     mov     [second], dx   ; leftover seconds
00006F65 29D2                   <1>     sub     edx, edx
00006F67 F7F1                   <1>     div     ecx
                                <1>     ;mov    [ihrs], eax    ; whole hours
                                <1>     ;                       ; since 1/1/1970
00006F69 668915[4C730100]       <1>     mov     [minute], dx   ; leftover minutes
00006F70 31D2                   <1>     xor     edx, edx
                                <1>     ;mov    cx, 24
00006F72 B118                   <1>     mov     cl, 24
00006F74 F7F1                   <1>     div     ecx
                                <1>     ;mov    [iday], ax    ; whole days
                                <1>                           ; since 1/1/1970
00006F76 668915[4A730100]       <1>     mov     [hour], dx    ; leftover hours
00006F7D 05DB020000             <1>     add     eax, 365+366 ; whole day since
                                <1>                          ; 1/1/1968
                                <1>     ;mov    [iday], ax
00006F82 50                     <1>     push    eax
00006F83 29D2                   <1>     sub     edx, edx
                                <1>     ;mov    ecx, (4*365)+1 ; 4 years = 1461 days
                                <1>     ; 25/07/2022
00006F85 66B9B505               <1>     mov     cx, (4*365)+1
00006F89 F7F1                   <1>     div     ecx
00006F8B 59                     <1>     pop     ecx
                                <1>     ;mov    [lday], ax    ; count of quadyrs (4 years)
                                <1>     ;push   dx
                                <1>     ; 18/04/2021
00006F8C 52                     <1>     push    edx
                                <1>     ;mov    [qday], dx    ; days since quadyr began
00006F8D 6683FA3C               <1>     cmp     dx, 31+29     ; if past feb 29 then
00006F91 F5                     <1>     cmc                   ; add this quadyr's leap day
00006F92 83D000                 <1>     adc     eax, 0        ; to # of qadyrs (leap days)
                                <1>     ;mov    [lday], ax    ; since 1968
                                <1>     ;mov    cx, [iday]
00006F95 91                     <1>     xchg    ecx, eax      ; ECX = lday, EAX = iday
00006F96 29C8                   <1>     sub     eax, ecx      ; iday - lday
                                <1>     ;mov    ecx, 365
                                <1>     ; 25/07/2022
00006F98 66B96D01               <1>     mov     cx, 365
00006F9C 31D2                   <1>     xor     edx, edx
                                <1>     ; EAX = iday-lday, EDX = 0
00006F9E 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
00006FA0 6605B007               <1>     add     ax, 1968      ; compute year
00006FA4 66A3[44730100]         <1>     mov     [year], ax
                                <1>     ;mov    cx, dx
                                <1>     ; 25/07/2022
00006FAA 89D1                   <1>     mov     ecx, edx
                                <1>     ;;mov   dx, [qday]
                                <1>     ;pop    dx
                                <1>     ; 18/04/2021
00006FAC 5A                     <1>     pop     edx
00006FAD 6681FA6D01             <1>     cmp     dx, 365      ; if qday <= 365 and qday >= 60
00006FB2 7708                   <1>     ja      short cfe1   ; jday = jday +1
00006FB4 6683FA3C               <1>     cmp     dx, 60       ; if past 2/29 and leap year then
00006FB8 F5                     <1>         cmc              ; add a leap day to the # of whole
                                <1>     ;adc    cx, 0        ; days since 1/1 of current year
                                <1>     ; 25/07/2022
00006FB9 83D100                 <1>     adc     ecx, 0
                                <1> cfe1:
                                <1>     ;mov    [jday], cx
                                <1>     ;mov    bx, 12       ; estimate month
                                <1>     ; 18/04/2021
00006FBC 29DB                   <1>     sub     ebx, ebx
00006FBE B30C                   <1>     mov     bl, 12
00006FC0 66BA6E01               <1>     mov     dx, 366      ; mday, max. days since 1/1 is 365
00006FC4 6683E003               <1>     and     ax, 11b      ; year mod 4 (and dx, 3)
                                <1> cfe2:       ; Month calculation  ; 0 to 11  (11 to 0)
                                <1>     ;cmp    cx, dx       ; mday = # of days passed from 1/1
                                <1>     ; 25/07/2022
00006FC8 39D1                   <1>     cmp     ecx, edx
00006FCA 731B                   <1>     jnb     short cfe3
                                <1>     ;dec    bx           ; month = month - 1
                                <1>     ;shl    bx, 1
                                <1>     ; 18/04/2021
00006FCC FECB                   <1>     dec     bl
00006FCE D0E3                   <1>     shl     bl, 1
00006FD0 668B93[50730100]       <1>     mov     dx, [EBX+DMonth] ; # elapsed days at 1st of month
                                <1>     ; 18/04/2021
                                <1>     ;shr    bx, 1        ; bx = month - 1 (0 to 11)
00006FD7 D0EB                   <1>     shr     bl, 1
                                <1>     ;cmp    bx, 1        ; if month > 2 and year mod 4  = 0
00006FD9 80FB01                 <1>     cmp     bl, 1
00006FDC 76EA                   <1>     jna     short cfe2   ; then mday = mday + 1
00006FDE 76E8                   <1>     jna     short cfe2   ; then mday = mday + 1
00006FE0 08C0                   <1>     or      al, al       ; if past 2/29 and leap year then
00006FE2 75E4                   <1>     jnz     short cfe2   ; add leap day (to mday)
                                <1>     ;inc    dx           ; mday = mday + 1
                                <1>     ; 25/07/2022
00006FE4 42                     <1>     inc     edx
00006FE5 EBE1                   <1>     jmp     short cfe2
                                <1> cfe3:
                                <1>     ;inc    bx          ; -> bx = month, 1 to 12
                                <1>     ; 18/04/2021
00006FE7 FEC3                   <1>     inc     bl
00006FE9 66891D[46730100]       <1>     mov     [month], bx
                                <1>     ;sub    cx, dx       ; day = jday - mday + 1
                                <1>     ; 25/07/2022
00006FF0 29D1                   <1>     sub     ecx, edx
                                <1>     ;inc    cx
                                <1>     ; 18/04/2021
```

```
697 00006FF2 FEC1                <1>     inc     cl
698                              <1>     ;mov    [day], cx
699 00006FF4 880D[48730100]      <1>     mov     [day], cl
700                              <1>
701                              <1>     ; eax, ebx, ecx, edx is changed at return
702                              <1>     ; output ->
703                              <1>     ; [year], [month], [day], [hour], [minute], [second]
704                              <1>
705 00006FFA C3                  <1>     retn    ; 31/12/2017 (TRDOS 386)
706                              <1>
707                              <1> set_rtc_date_time:
708                              <1>     ; 31/12/2017 (v2.0.0)
709                              <1>     ; 30/12/2017 (TRDOS 386)
710                              <1>     ; 15/03/2015 (Retro UNIX 386 v1 - 32 bit version)
711                              <1>     ; 20/06/2013 (Retro UNIX 8086 v1)
712 00006FFB E80F000000          <1>     call    set_date_bcd
713                              <1>     ; Set real-time clock date
714 00007000 E8DDF3FFFF          <1>     call    set_rtc_date ; RTC_50
715                              <1>     ; Set real-time clock time
716 00007005 E832000000          <1>     call    set_time_bcd
717 0000700A E970F3FFFF          <1>     jmp     set_rtc_time ; RTC_30
718                              <1>
719                              <1> ; 31/12/2017
720                              <1> set_date_bcd:
721 0000700F A0[45730100]        <1>         mov     al, [year+1]
722 00007014 D40A                <1>     aam     ; ah = al / 10, al = al mod 10
723 00007016 D510                <1>     db      0D5h, 10h    ; Undocumented inst. AAD
724                              <1>                          ; AL = AH * 10h + AL
725 00007018 88C5                <1>     mov     ch, al ; century (BCD)
726 0000701A A0[44730100]        <1>     mov     al, [year]
727 0000701F D40A                <1>     aam     ; ah = al / 10, al = al mod 10
728 00007021 D510                <1>     db      0D5h, 10h    ; Undocumented inst. AAD
729                              <1>                          ; AL = AH * 10h + AL
730 00007023 88C1                <1>     mov     cl, al ; year (BCD)
731 00007025 A0[46730100]        <1>         mov         al, [month]
732 0000702A D40A                <1>     aam     ; ah = al / 10, al = al mod 10
733 0000702C D510                <1>     db      0D5h, 10h    ; Undocumented inst. AAD
734                              <1>                          ; AL = AH * 10h + AL
735 0000702E 88C6                <1>     mov     dh, al ; month (BCD)
736 00007030 A0[48730100]        <1>     mov     al, [day]
737 00007035 D40A                <1>     aam     ; ah = al / 10, al = al mod 10
738 00007037 D510                <1>     db      0D5h, 10h    ; Undocumented inst. AAD
739                              <1>                          ; AL = AH * 10h + AL
740                              <1>     ; 18/04/2021
741 00007039 88C2                <1>     mov     dl, al ; day (BCD)
742 0000703B C3                  <1>     retn    ; 30/12/2017
743                              <1>
744                              <1> ; 31/12/2017
745                              <1> set_time_bcd:
746                              <1>         ; Read real-time clock time
747                              <1>     ; (get day light saving time bit status)
748 0000703C FA                  <1>     cli
749 0000703D E8DBF4FFFF          <1>     call    UPD_IPR              ; CHECK FOR UPDATE IN PROCESS
750                              <1>     ; cf = 1 -> al = 0
751 00007042 7207                <1>         jc      short stime1
752 00007044 B00B                <1>     mov     al, CMOS_REG_B       ; ADDRESS ALARM REGISTER
753 00007046 E808F5FFFF          <1>     call    CMOS_READ            ; READ CURRENT VALUE OF DSE BIT
754                              <1> stime1:
755 0000704B FB                  <1>     sti
756 0000704C 2401                <1>     and     al, 00000001b        ; MASK FOR VALID DSE BIT
757 0000704E 88C2                <1>     mov     dl, al               ; SET [DL] TO ZERO FOR NO DSE BIT
758                              <1>     ; DL = 1 or 0 (day light saving time)
759                              <1>
760 00007050 A0[4A730100]        <1>     mov     al, [hour]
761 00007055 D40A                <1>     aam     ; ah = al / 10, al = al mod 10
762 00007057 D510                <1>     db      0D5h,10h     ; Undocumented inst. AAD
763                              <1>                          ; AL = AH * 10h + AL
764 00007059 88C5                <1>     mov     ch, al ; hour (BCD)
765 0000705B A0[4C730100]        <1>         mov     al, [minute]
766 00007060 D40A                <1>     aam     ; ah = al / 10, al = al mod 10
767 00007062 D510                <1>     db      0D5h,10h     ; Undocumented inst. AAD
768                              <1>                          ; AL = AH * 10h + AL
769 00007064 88C1                <1>     mov     cl, al       ; minute (BCD)
770 00007066 A0[4E730100]        <1>         mov     al, [second]
771 0000706B D40A                <1>     aam     ; ah = al / 10, al = al mod 10
772 0000706D D510                <1>     db      0D5h,10h     ; Undocumented inst. AAD
773                              <1>                          ; AL = AH * 10h + AL
774 0000706F 88C6                <1>     mov     dh, al       ; second (BCD)
775 00007071 C3                  <1>     retn    ; 30/12/2017
3433                                 %include 'trdosk2.s' ; 04/01/2016
   1                             <1> ; ****************************************************************************
   2                             <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.10) - DRV INIT : trdosk2.s
   3                             <1> ; ----------------------------------------------------------------------------
   4                             <1> ; Last Update: 19/12/2025 (Previous: 22/05/2024, TRDOS 386 v2.0.8)
   5                             <1> ; ----------------------------------------------------------------------------
   6                             <1> ; Beginning: 04/01/2016
   7                             <1> ; ----------------------------------------------------------------------------
   8                             <1> ; Assembler: NASM version 2.14 (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> ; DRV_INIT.ASM (c) 2009-2011 Erdogan TAN  [26/09/2009] Last Update: 07/08/2011
  14                             <1> ;
  15                             <1>
  16                             <1> ldrv_init: ; Logical Drive Initialization
  17                             <1>     ; 19/12/2025 (TRDOS 386 Kernel v2.0.10)
  18                             <1>     ; 22/05/2024 (TRDOS 386 Kernel v2.0.8)
  19                             <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
  20                             <1>     ; 30/08/2020
  21                             <1>     ; 25/08/2020
  22                             <1>     ; 11/08/2020 - 13/08/2020
  23                             <1>     ; 17/07/2020 - 20/07/2020
  24                             <1>     ; 14/07/2020 - 15/07/2020
  25                             <1>     ; 30/01/2018
  26                             <1>     ; 27/12/2017
  27                             <1>     ; 12/02/2016
  28                             <1>     ; 06/01/2016
  29                             <1>     ;       ('diskinit.inc', 'diskio.inc' integration)
  30                             <1>     ; 04/01/2016 (TRDOS 386 = TRDOS v2.0)
  31                             <1>     ; 07/08/2011
  32                             <1>     ; 20/09/2009
  33                             <1>     ; 2005
  34                             <1>
  35                             <1>     ; 15/07/2020
  36                             <1>     ;movzx  ecx, byte [HF_NUM] ; number of fixed disks
  37                             <1>     ;cmp    cl, 1
  38                             <1>     ;jnb    short load_hd_partition_tables
  39                             <1>
  40 00007072 A0[18770100]       <1>     mov     al, [HF_NUM] ; number of fixed disks
  41 00007077 20C0               <1>     and     al, al
  42 00007079 7501               <1>     jnz     short load_hd_partition_tables
  43                             <1>
  44                             <1>     ; no any hard disks
```

```
0000707B C3                 <1>     retn
                            <1>
                            <1> load_hd_partition_tables:
                            <1>     ;mov    esi, [HDPM_TBL_VEC] ; primary master disk FDPT
                            <1>     ; 15/07/2020
0000707C BE[1C770100]       <1>     mov     esi, HDPM_TBL_VEC
00007081 BF[427B0100]       <1>     mov     edi, PTable_hd0
00007086 B280               <1>     mov     dl, 80h
                            <1>     ; 15/07/2020
00007088 A2[7B660000]       <1>     mov     [hdc], al
                            <1>     ;xor    ecx, ecx ; 0
                            <1> load_next_hd_partition_table:
                            <1>     ; 20/07/2020
0000708D 31C9               <1>     xor     ecx, ecx ; 0
                            <1>     ;push   ecx
0000708F 57                 <1>     push    edi ; *
                            <1>     ;push   esi ; FDPT (+ DPTE) address
                            <1>     ; 15/07/2020
00007090 AD                 <1>     lodsd
00007091 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
00007092 8A4814             <1>     mov     cl, [eax+20]
00007095 80E140             <1>     and     cl, 40h
                            <1>     ;mov    [HD_LBA_yes], cl
                            <1>     ; 22/05/2024 (BugFix)
00007098 0FB6C2             <1>     movzx   eax, dl
0000709B 05[C27B0100]       <1>     add     eax, HD_LBA_yes - 80h
000070A0 8808               <1>     mov     [eax], cl
                            <1>
000070A2 E81A030000         <1>     call    load_masterboot
                            <1>     ;jc     short pass_pt_this_hard_disk
                            <1>     ; 13/08/2020
                            <1>     ;jc     pass_pt_this_hard_disk
                            <1>     ; 25/07/2022
000070A7 7302               <1>     jnc     short load_mbr_ok
000070A9 EB7C               <1>     jmp     pass_pt_this_hard_disk
                            <1>
                            <1> load_mbr_ok:
000070AB BB[007B0100]       <1>     mov     ebx, PartitionTable
000070B0 89DE               <1>     mov     esi, ebx
                            <1>     ;mov    ecx, 16
000070B2 B110               <1>     mov     cl, 16
000070B4 F3A5               <1>     rep     movsd
000070B6 89DE               <1>     mov     esi, ebx
                            <1>     ;mov    byte [hdc], 4 ; 4 - partition index
                            <1>     ; 15/07/2020
000070B8 C605[477C0100]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
000070BF 8A4604             <1>     mov     al, [esi+ptFileSystemID]
000070C2 20C0               <1>     and     al, al
000070C4 7449               <1>     jz      short loc_validate_next_hdp_partition2
                            <1>
000070C6 56                 <1>     push    esi ; *** ; Masterboot partition table offset
000070C7 52                 <1>     push    edx ; **** ; dl = Physical drive number
                            <1>
                            <1>     ; 13/08/2020
000070C8 3C05               <1>     cmp     al, 05h  ; Extended partition CHS
000070CA 7404               <1>     je      short loc_set_ep_counter
000070CC 3C0F               <1>     cmp     al, 0Fh  ; Extended partition LBA
000070CE 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
000070D0 803D[487C0100]80   <1>     cmp     byte [EP_Counter], 80h
000070D7 7334               <1>     jnb     short loc_validate_next_hdp_partition1
                            <1>
000070D9 8815[487C0100]     <1>     mov     byte [EP_Counter], dl ; disk drv has extd. part.
                            <1>
000070DF EB2C               <1>     jmp     short loc_validate_next_hdp_partition1
                            <1>
                            <1> loc_validate_next_hdp_partition0:
000070E1 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
000070E3 E87E010000         <1>     call    validate_hd_fat_partition
                            <1>
                            <1> ; 19/12/2025
                            <1> %if 0
                            <1>     jnc     short loc_set_valid_hdp_partition_entry
                            <1>
                            <1>     ;pop    edx
                            <1>     ;push   edx
                            <1>     mov     edx, [esp]  ; ****
                            <1>     mov     esi, [esp+4] ; *** ; 30/01/2018
                            <1>     call    validate_hd_fs_partition
                            <1> %endif
000070E8 7223               <1>     jc      short loc_validate_next_hdp_partition1
                            <1>
                            <1> loc_set_valid_hdp_partition_entry:
000070EA 8A0D[5C2E0100]     <1>     mov     cl, [Last_DOS_DiskNo]
000070F0 80C141             <1>     add     cl, 'A'
                            <1>     ; ESI = Logical dos drive description table address
000070F3 880E               <1>     mov     [esi+LD_Name], cl
                            <1>     ; 15/07/2020
000070F5 8A4602             <1>     mov     al, [esi+LD_PhyDrvNo] ; Physical drive number
                            <1>     ;mov    al, [esp] ; ****
000070F8 2C7F               <1>     sub     al, 7Fh
                            <1>             ; AL = 1 to 4
000070FA C0E002             <1>     shl     al, 2 ; AL = 4 to 16
                            <1>
000070FD 8A15[477C0100]     <1>     mov     dl, [PP_Counter]
                            <1>
                            <1>     ;sub    al, [PP_Counter]
00007103 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
00007105 B404               <1>     mov     ah, 4
                            <1>     ;sub    ah, [PP_Counter]
00007107 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
00007109 6689467C           <1>     mov     [esi+LD_PartitionEntry], ax
                            <1>
                            <1> loc_validate_next_hdp_partition1:
0000710D 5A                 <1>     pop     edx ; **** ; dl = Physical drive number
0000710E 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
0000710F FE0D[477C0100]     <1>     dec     byte [PP_Counter] ; 4 - partition index
00007115 7410               <1>     jz      short pass_pt_this_hard_disk
                            <1>
00007117 83C610             <1>     add     esi, 16 ; 10h
0000711A EBA3               <1>     jmp     short loc_validate_hdp_partition
                            <1>
                            <1> loc_not_any_extd_partitions:
                            <1>     ; 15/07/2020
0000711C C3                 <1>     retn
                            <1>
                            <1> loc_next_hd_partition_table:
0000711D FEC2               <1>     inc     dl
                            <1>     ; 15/07/2020
                            <1>     ;add    esi, 32 ; next FDPT address
0000711F 83C740             <1>     add     edi, 64 ; next partition table destination
00007122 E966FFFFFF         <1>         jmp     load_next_hd_partition_table
                            <1>
                            <1> pass_pt_this_hard_disk:
                            <1>     ;pop    esi ; FDPT (+ DPTE) address
                            <1>     ; 15/07/2020
00007127 5E                 <1>     pop     esi ; ** ; next FDPT (+ DPTE) address ptr
00007128 5F                 <1>     pop     edi ; * ; Ptable_hd?
                            <1>     ;pop    ecx
                            <1>     ;loop   loc_next_hd_partition_table
00007129 FE0D[7B660000]     <1>     dec     byte [hdc]
0000712F 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
00007131 A0[487C0100]       <1>     mov     al, [EP_Counter] ; 1st disk drv has extd partition
00007136 08C0               <1>     or      al, al ; 0 ?
00007138 74E2               <1>     jz      short loc_not_any_extd_partitions
                            <1>
                            <1> load_extended_dos_partitions:
                            <1>     ;mov    byte [hdc], 80h
                            <1>     ; 13/08/2020
0000713A A2[7B660000]       <1>     mov     byte [hdc], al ; 1st disk drv has extd partition
                            <1>     ; 25/08/2020
0000713F 2C80               <1>     sub     al, 80h
00007141 740E               <1>     jz      short loc_set_ext_ptable_hd0
00007143 C0E006             <1>     shl     al, 6 ; * 64
00007146 0FB6F0             <1>     movzx   esi, al
00007149 81C6[427B0100]     <1>     add     esi, PTable_hd0
0000714F EB05               <1>     jmp     short next_hd_extd_partition
                            <1>
                            <1>     ; 25/08/2020
                            <1> loc_set_ext_ptable_hd0:
00007151 BE[427B0100]       <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
00007156 66C705[487C0100]00- <1>     mov     word [EP_Counter], 0 ; Reset EP index and LD index
0000715E 00                 <1>
                            <1>
0000715F 56                 <1>     push    esi ; **** ; PTable_hd? offset
                            <1>
00007160 C605[477C0100]04   <1>     mov     byte [PP_Counter], 4
                            <1>                             ; set for each extd partition table
                            <1>     ;;mov   ecx, 4
                            <1>     ;mov    cl, 4
00007167 8A15[7B660000]     <1>     mov     dl, [hdc]
                            <1> hd_check_fs_id_05h:
0000716D 8A4604             <1>     mov     al, [esi+ptFileSystemID]
00007170 3C05               <1>     cmp     al, 05h ; Is it an extended dos partition ?
00007172 7411               <1>     je      short loc_set_ep_start_sector ; yes
                            <1> hd_check_fs_id_0Fh:
00007174 3C0F               <1>     cmp     al, 0Fh ; Is it an extended win4 (LBA mode) partition ?
00007176 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
00007178 FE0D[477C0100]     <1>     dec     byte [PP_Counter] ; 4 --> 0
0000717E 7432               <1>     jz      short continue_check_ep_next_disk
00007180 83C610             <1>     add     esi, 16
00007183 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
00007185 8B4E08             <1>     mov     ecx, [esi+ptStartSector]
                            <1>     ; 30/08/2020
00007188 890D[4A7C0100]     <1>     mov     [MBR_EP_StartSector], ecx
                            <1>     ; 20/07/2020
```

```
                            <1> loc_validate_hde_partition_next:
                            <1>     ; 22/05/2024 (BugFix)
0000718E 0FB6FA             <1>     movzx   edi, dl
00007191 81C7[C27B0100]     <1>     add     edi, HD_LBA_yes - 80h
                            <1>     ;
00007197 890D[4E7C0100]     <1>     mov     [EP_StartSector], ecx ; Extended partition's start sector
0000719D BB[42790100]       <1>         mov ebx, MasterBootBuff
                            <1>     ; 22/05/2024
000071A2 803F01             <1>     cmp     byte [edi], 1 ; LBA ready = Yes
                            <1>     ;cmp    byte [HD_LBA_yes], 1 ; LBA ready = Yes
000071A5 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
000071A7 66B8011B           <1>     mov     ax, 1B01h
000071AB E85BDEFFFF         <1>     call    int13h
                            <1>     ;pop    ecx
                            <1>     ;jnc    short loc_hd_move_ep_table
                            <1>     ; 15/07/2020
000071B0 732E               <1>     jnc     short loc_validate_hde_partition
                            <1>
                            <1> continue_check_ep_next_disk:
                            <1>     ; 15/07/2020
000071B2 5E                 <1>     pop     esi ; **** ; PTable_hd?
000071B3 A0[18770100]       <1>     mov     al, [HF_NUM] ; number of hard disks
000071B8 047F               <1>     add     al, 7Fh
000071BA 3805[7B660000]     <1>     cmp     [hdc], al
000071C0 730B               <1>     jnb     short loc_validating_hd_partitions_ok
000071C2 83C640             <1>     add     esi, 64
                            <1>     ; 15/07/2020
                            <1>     ;add    edi, 64
000071C5 FE05[7B660000]     <1>     inc     byte [hdc]
000071CB EB89               <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:
000071CD 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
000071CE 8A7601             <1>     mov     dh, [esi+ptBeginHead]
000071D1 668B4E02           <1>         mov     cx, [esi+ptBeginSector]
000071D5 66B80102           <1>     mov     ax, 0201h ; Read 1 sector
                            <1>     ;mov    ebx, MasterBootBuff
000071D9 E82DDEFFFF         <1>     call    int13h ; 20/07/2020
                            <1>                    ; 'diskio.s' modification, 'clc'
                            <1>     ;pop    ecx
000071DE 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
000071E0 BE[007B0100]       <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
000071E5 807E0400           <1>     cmp     byte [esi+ptFileSystemID], 0
000071E9 76C7               <1>     jna     short continue_check_ep_next_disk ; 13/08/2020
                            <1>
```

```
                            <1> loc_validate_minidisk_partition:
                            <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
                            <1>     ; 13/08/2020
                            <1>     ; 20/07/2020
                            <1>     ;push  esi ; *** ; Extended partition table offset
                            <1>
                            <1>     ; 13/08/2020
000071EB FE05[487C0100]     <1>     inc    byte [EP_Counter] ; current (sub partition) index
                            <1>                              ; in current extended partition
000071F1 BF[4E7C0100]       <1>     mov    edi, EP_StartSector
                            <1>
                            <1>     ; Input -> ESI = PartitionTable offset
                            <1>     ; DL = Hard disk drive number
                            <1>     ; EDI = Extended partition start sector pointer
000071F6 E86B000000         <1>     call   validate_hd_fat_partition
                            <1>     ;pop    ecx ; *
000071FB 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
000071FD 8A15[7B660000]     <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
00007203 EB3E               <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
00007205 A0[7B660000]       <1>     mov    al, [hdc] ; Hard disk drive number (>=80h)
0000720A 88C2               <1>     mov    dl, al ; mov dl, [hdc]
0000720C 2C7F               <1>     sub    al, 7Fh
                            <1>                    ; 1 to 4
0000720E C0E002             <1>     shl    al, 2 ; 4 to 16
00007211 2A05[477C0100]     <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
00007217 8A25[497C0100]     <1>     mov    ah, [LD_Counter] ; Logical drive index number
                            <1>                             ; (in current extended partition)
0000721D 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
00007220 6689467C           <1>     mov    [esi+LD_PartitionEntry], ax
                            <1>
00007224 8A0D[5C2E0100]     <1>     mov    cl, [Last_DOS_DiskNo]
0000722A 80C141             <1>     add    cl, 'A'
0000722D 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
0000722F FE05[497C0100]     <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
00007235 803D[497C0100]04   <1>     cmp    byte [LD_Counter], 4 ; maximum 4 logical disks
                            <1>                                 ; per extended partition
                            <1>     ;jnb    continue_check_ep_next_disk
                            <1>     ; 25/07/2022
0000723C 7205               <1>     jb     short validate_next_minidisk_partition_ok
0000723E E96FFFFFFF         <1>     jmp    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
00007243 BE[107B0100]       <1>     mov     esi, PartitionTable+16
                            <1>
                            <1>     ; 20/07/2020
00007248 8A4604             <1>     mov     al, [esi+ptFileSystemID]
                            <1>
                            <1>     ; 20/07/2020
0000724B 3C05               <1>     cmp     al, 05h ; Is it an extended dos partition ?
0000724D 7409               <1>     je      short loc_minidisk_next_ep_lba_chs ; 17/07/2020
0000724F 3C0F               <1>     cmp     al, 0Fh ; Is it an extended win4 (LBA mode) partition ?
                            <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>     ; 25/07/2022
00007251 7405               <1>     je      short loc_minidisk_next_ep_lba_chs
00007253 E95AFFFFFF         <1>     jmp     continue_check_ep_next_disk
                            <1>
                            <1> loc_minidisk_next_ep_lba_chs:
                            <1>     ; 17/07/2020
00007258 8B4E08             <1>     mov     ecx, [esi+ptStartSector] ; relative start sector number
                            <1>     ;add    ecx, [EP_StartSector]
                            <1>     ; 30/08/2020
0000725B 030D[4A7C0100]     <1>     add     ecx, [MBR_EP_StartSector]
                            <1>     ; 20/07/2020
00007261 E928FFFFFF         <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
00007266 8A6604             <1>     mov     ah, [esi+ptFileSystemID]
00007269 B002               <1>     mov     al, 2 ; 27/12/2017
0000726B 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)
0000726E 7711               <1>     ja      short vhdp_FAT16_32
00007270 7425               <1>     je      short loc_set_valid_hd_partition_params
                            <1>
                            <1> vhdp_FAT12_16:
                            <1>     ; 27/12/2017
00007272 FEC8               <1>     dec     al ; mov al, 1
00007274 38C4               <1>     cmp     ah, al ; 1 ; FAT12 partition
00007276 741F               <1>     je      short loc_set_valid_hd_partition_params
                            <1>     ;
00007278 FEC0               <1>     inc     al ; mov al, 2
0000727A 80FC04             <1>     cmp     ah, 04h ; FAT16 CHS partition (< 32MB)
0000727D 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:
0000727F F9                 <1>     stc
                            <1>     ; cf=1
00007280 C3                 <1>     retn
                            <1>
                            <1> vhdp_FAT16_32:
                            <1>     ; 15/07/2020
                            <1>     ;mov    al, 3
00007281 FEC0               <1>     inc     al
00007283 80FC0C             <1>     cmp     ah, 0Ch ; FAT32 LBA partition
00007286 740F               <1>     je      short loc_set_valid_hd_partition_params
00007288 7706               <1>     ja      short vhdp_check_FAT16_lba
                            <1>
                            <1> vhdp_check_FAT32_chs:
0000728A 80FC0B             <1>     cmp     ah, 0Bh ; FAT32 CHS partition
0000728D 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:
0000728F C3                 <1>     retn
                            <1>
                            <1> vhdp_check_FAT16_lba:
00007290 80FC0E             <1>     cmp     ah, 0Eh ; FAT16 LBA partition
00007293 75EA               <1>     jne     short loc_not_a_valid_fat_partition1
                            <1>
                            <1>     ;mov    al, 2
00007295 FEC8               <1>     dec     al
                            <1>
                            <1> loc_set_valid_hd_partition_params:
                            <1>     ; 19/12/2025 (TRDOS 386 Kernel v2.0.10)
                            <1>     ; 30/07/2022
                            <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
                            <1>     ; 15/07/2020
                            <1>     ;inc    byte [Last_DOS_DiskNo] ; > 1
                            <1>     ;
00007297 31DB               <1>     xor     ebx, ebx
00007299 8A3D[5C2E0100]     <1>     mov     bh, [Last_DOS_DiskNo] ; * 256
0000729F FEC7               <1>     inc     bh ; 15/07/2020
000072A1 81C300010900       <1>     add     ebx, Logical_DOSDisks
                            <1>     ;
000072A7 C6430102           <1>     mov     byte [ebx+LD_DiskType], 2
000072AB 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
```

```
000072AE 66894303            <1>     mov     word [ebx+LD_FATType], ax
                             <1>     ;
000072B2 8B4E08              <1>     mov     ecx, [esi+ptStartSector]
000072B5 09FF                <1>     or      edi, edi
000072B7 7402                <1>     jz      short pass_hd_FAT_ep_start_sector_adding
                             <1> loc_add_hd_FAT_ep_start_sector:
                             <1>     ; 17/07/2020
000072B9 030F                <1>     add     ecx, [edi]
                             <1> pass_hd_FAT_ep_start_sector_adding:
000072BB 894B6C              <1>     mov     [ebx+LD_StartSector], ecx
                             <1> loc_hd_FAT_logical_drv_init:
000072BE 89DD                <1>     mov     ebp, ebx
                             <1>     ;mov    dl, [ebx+LD_PhyDrvNo]
000072C0 A0[427C0100]        <1>     mov     al, [HD_LBA_yes] ; 07/01/2016
000072C5 884305              <1>     mov     [ebx+LD_LBAYes], al
000072C8 BB[527C0100]        <1>     mov     ebx, DOSBootSectorBuff ; buffer address
000072CD 08C0                <1>     or      al, al
000072CF 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)
000072D1 B41B                <1>     mov     ah, 1Bh ; LBA read
000072D3 B001                <1>     mov     al, 1 ; sector count
000072D5 E831DDFFFF          <1>     call    int13h
000072DA 7313                <1>     jnc     short loc_hd_drv_FAT_boot_validation
                             <1> loc_not_a_valid_fat_partition3:
000072DC C3                  <1>     retn
                             <1> loc_hd_FAT_drv_init_load_bs_chs:
000072DD 8A7601              <1>     mov     dh, [esi+ptBeginHead]
000072E0 668B4E02            <1>     mov     cx, [esi+ptBeginSector]
000072E4 66B80102            <1>     mov     ax, 0201h ; Read 1 sector
                             <1>     ;mov    ebx, DOSBootSectorBuff
000072E8 E81EDDFFFF          <1>     call    int13h
000072ED 72ED                <1>     jc      short loc_not_a_valid_fat_partition3
                             <1> loc_hd_drv_FAT_boot_validation:
                             <1>     ;mov    esi, DOSBootSectorBuff
000072EF 89DE                <1>     mov     esi, ebx
000072F1 6681BEFE01000055AA  <1>     cmp     word [esi+BS_Validation], 0AA55h
000072FA 751C                <1>     jne     short loc_not_a_valid_fat_partition4
000072FC 807E15F8            <1>     cmp     byte [esi+BPB_Media], 0F8h
00007300 7516                <1>     jne     short loc_not_a_valid_fat_partition4
                             <1>     ; 19/12/2025
00007302 66817E0B0002        <1>     cmp     word [esi+BPB_BytsPerSec], 512
00007308 750E                <1>     jne     short loc_not_a_valid_fat_partition4
                             <1>
                             <1>     ; 25/07/2022
0000730A 31C9                <1>     xor     ecx, ecx
                             <1>
                             <1>     ; 27/12/2017
0000730C 807D0303            <1>     cmp     byte [ebp+LD_FATType], 3
00007310 7508                <1>     jne     short loc_hd_FAT16_BPB
                             <1>
                             <1> loc_hd_drv_FAT32_boot_validation:
00007312 807E4229            <1>     cmp     byte [esi+BS_FAT32_BootSig], 29h
00007316 7413                <1>     je      short loc_hd_FAT32_BPB
                             <1>
                             <1> loc_not_a_valid_fat_partition4:
00007318 F9                  <1>     stc
00007319 C3                  <1>     retn
                             <1>
                             <1> loc_hd_FAT16_BPB:
0000731A 807E2629            <1>     cmp     byte [esi+BS_BootSig], 29h
0000731E 75F8                <1>     jne     short loc_not_a_valid_fat_partition4
                             <1>
00007320 66837E1600          <1>     cmp     word [esi+BPB_FATSz16], 0
00007325 7604                <1>     jna     short loc_hd_big_FAT16_BPB
                             <1>     ;mov    ecx, 32
                             <1>     ; 25/07/2022
00007327 B120                <1>     mov     cl, 32
                             <1>     ; ecx = 32
00007329 EB02                <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:
                             <1>     ;mov    ecx, 45
                             <1>     ; 25/07/2022
0000732B B12D                <1>     mov     cl, 45
                             <1>     ; ecx = 45
                             <1> loc_hd_move_FAT_BPB:
0000732D 89EF                <1>     mov     edi, ebp
                             <1>     ;mov    esi, ebx ; Boot sector
0000732F 57                  <1>     push    edi
00007330 83C706              <1>     add     edi, LD_BPB
00007333 F366A5              <1>     rep     movsw
00007336 5E                  <1>     pop     esi
00007337 0FB74614            <1>     movzx   eax, word [esi+LD_BPB+BPB_RsvdSecCnt]
0000733B 03466C              <1>     add     eax, [esi+LD_StartSector]
0000733E 894660              <1>     mov     [esi+LD_FATBegin], eax
00007341 807E0303            <1>     cmp     byte [esi+LD_FATType], 3
00007345 7223                <1>     jb      short loc_set_FAT16_RootDirLoc
                             <1> loc_set_FAT32_RootDirLoc:
00007347 8B462A              <1>     mov     eax, [esi+LD_BPB+BPB_FATSz32]
0000734A 0FB65E16            <1>         movzx       ebx, byte [esi+LD_BPB+BPB_NumFATs]
0000734E F7E3                <1>     mul     ebx
00007350 034660              <1>     add     eax, [esi+LD_FATBegin]
                             <1> loc_set_FAT32_data_begin:
00007353 894668              <1>     mov     [esi+LD_DATABegin], eax
00007356 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.
00007359 8B4632              <1>     mov     eax, [esi+LD_BPB+BPB_RootClus]
                             <1>     ;sub    eax, 2
                             <1>     ; 30/07/2022
0000735C 48                  <1>     dec     eax ; 2 -> 1
0000735D 48                  <1>     dec     eax ; 1 -> 0
0000735E 7433                <1>     jz      short short loc_set_32bit_FAT_total_sectors
                             <1>     ;movzx  ebx, byte [esi+LD_BPB+BPB_SecPerClust]
00007360 8A5E13              <1>     mov     bl, [esi+LD_BPB+BPB_SecPerClust]
00007363 F7E3                <1>     mul     ebx
00007365 014664              <1>     add     [esi+LD_ROOTBegin], eax
00007368 EB29                <1>     jmp     short loc_set_32bit_FAT_total_sectors
                             <1>     ;
                             <1> loc_set_FAT16_RootDirLoc:
0000736A 0FB64616            <1>     movzx   eax, byte [esi+LD_BPB+BPB_NumFATs]
0000736E 0FB7561C            <1>     movzx   edx, word [esi+LD_BPB+BPB_FATSz16]
00007372 F7E2                <1>     mul     edx
00007374 034660              <1>     add     eax, [esi+LD_FATBegin]
```

```
00007377 894664              <1>     mov     [esi+LD_ROOTBegin], eax
                             <1> loc_set_FAT16_data_begin:
0000737A 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
0000737D 0FB74617            <1>     movzx   eax, word [esi+LD_BPB+BPB_RootEntCnt]
00007381 6683C00F            <1>     add     ax, 15
                             <1>     ;shr    ax, 4 ; / 16 ; (16 entries per sector)
                             <1>     ; 25/07/2022
00007385 C1E804              <1>     shr     eax, 4
00007388 014668              <1>     add     [esi+LD_DATABegin], eax
                             <1>     ;movzx  eax, word [esi+LD_BPB+BPB_TotalSec16]
0000738B 668B4619            <1>     mov     ax, [esi+LD_BPB+BPB_TotalSec16]
                             <1>     ;test   ax, ax
                             <1>     ; 25/07/2022
0000738F 85C0                <1>     test    eax, eax
                             <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
00007391 7503                <1>     jnz     short loc_set_hd_FAT_cluster_count
                             <1> loc_set_32bit_FAT_total_sectors:
00007393 8B4626              <1>     mov     eax, [esi+LD_BPB+BPB_TotalSec32]
                             <1>     ;mov    [esi+LD_TotalSectors], eax
                             <1> loc_set_hd_FAT_cluster_count:
00007396 894670              <1>     mov     [esi+LD_TotalSectors], eax ; 14/07/2020
00007399 8B5668              <1>     mov     edx, [esi+LD_DATABegin]
0000739C 2B566C              <1>     sub     edx, [esi+LD_StartSector]
0000739F 29D0                <1>     sub     eax, edx
000073A1 31D2                <1>     xor     edx, edx ; 0
000073A3 0FB64E13            <1>         movzx   ecx, byte [esi+LD_BPB+BPB_SecPerClust]
000073A7 F7F1                <1>         div ecx
000073A9 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
000073AC E831000000          <1>     call    get_free_FAT_sectors
000073B1 720D                <1>     jc      short loc_validate_hd_FAT_partition_retn
000073B3 894674              <1>     mov     [esi+LD_FreeSectors], eax
000073B6 C6467E06            <1>     mov     byte [esi+LD_MediaChanged], 6  ; Volume Name Reset
                             <1>
                             <1>     ; 15/07/2020
000073BA FE05[5C2E0100]      <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:
000073C0 C3                  <1>     retn
                             <1>
                             <1> ; 19/12/2025
                             <1> %if 0
                             <1>
                             <1> validate_hd_fs_partition:
                             <1>     ; 19/12/2025 (TRDOS 386 v2.0.10)
                             <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
                             <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
                             <1>     mov     ah, [esi+ptFileSystemID]
                             <1>     cmp     ah, 0A1h ; SINGLIX FS1 (trfs1) partition
                             <1>     jne     short loc_validate_hd_fs_partition_stc_retn
                             <1> loc_set_valid_hd_fs_partition_params:
                             <1>     inc     byte [Last_DOS_DiskNo] ; > 1
                             <1>     xor     al, al ; mov al, 0
                             <1>     ;mov    [drv], dl
                             <1>     sub     ebx, ebx ; 0
                             <1>     mov     bh, [Last_DOS_DiskNo]
                             <1>     add     ebx, Logical_DOSDisks
                             <1>     mov     byte [ebx+LD_DiskType], 2
                             <1>     mov     [ebx+LD_PhyDrvNo], dl
                             <1>     ;mov    [ebx+LD_FATType], al ; 0
                             <1>     ;mov    [ebx+LD_FSType], ah
                             <1>     mov     [ebx+LD_FATType], ax
                             <1>     ;mov    eax, [esi+ptStartSector]
                             <1>     ;mov    [ebx+LD_StartSector], eax
                             <1> loc_hd_fs_logical_drv_init:
                             <1>     mov     ebp, ebx ; 10/01/2016
                             <1>     ;mov    dl, [ebx+LD_PhyDrvNo]
                             <1>     mov     al, [HD_LBA_yes] ; 10/01/2016
                             <1>     mov     [ebx+LD_LBAYes], al
                             <1>     mov     esi, ebx
                             <1>     mov     ebx, DOSBootSectorBuff ; buffer address
                             <1>     or      al, al
                             <1>     jnz     short loc_hd_fs_drv_init_load_bs_lba
                             <1> loc_hd_fs_drv_init_load_bs_chs:
                             <1>     mov     dh, [esi+ptBeginHead]
                             <1>     mov     cx, [esi+ptBeginSector]
                             <1>     mov     ax, 0201h ; Read 1 sector
                             <1>     ;mov    ebx, DOSBootSectorBuff
                             <1>     call    int13h
                             <1>     jnc     short loc_hd_drv_fs_boot_validation
                             <1> loc_validate_hd_fs_partition_err_retn:
                             <1>     retn
                             <1> loc_validate_hd_fs_partition_stc_retn:
                             <1>     stc
                             <1>     retn
```

```
<1> loc_hd_fs_drv_init_load_bs_lba:
<1>     ; DL = Physical drive number
<1>     ;mov    esi, ebx
<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)
<1>     mov     ah, 1Bh ; LBA read
<1>     mov     al, 1 ; sector count
<1>     call    int13h
<1>     jc      short loc_validate_hd_fs_partition_err_retn
<1> loc_hd_drv_fs_boot_validation:
<1>     ;mov    esi, DOSBootSectorBuff
<1>     mov     esi, ebx ; Boot sector buffer
<1>     cmp     word [esi+BS_Validation], 0AA55h
<1>     jne     short loc_validate_hd_fs_partition_stc_retn
<1>         ;
<1>     ;Singlix FS Extensions to TR-DOS (7/6/2009)
<1>     cmp     word [esi+bs_FS_Identifier], 'FS' ; 03/02/2018
<1>     jne     short loc_validate_hd_fs_partition_stc_retn
<1>        ;'A1h' check is not necessary
<1>     ;  if 'FS' check is passed as OK/Yes.
<1>     cmp     byte [esi+bs_FS_PartitionID], 0A1h
<1>     jne     short loc_validate_hd_fs_partition_stc_retn
<1>     ;
<1>     mov     edi, ebp ; 10/01/2016
<1>     ;
<1>     mov     al, byte [esi+bs_FS_LBA_Ready]
<1>     mov     [edi+LD_FS_LBAYes], al
<1>     ;
<1>     ; 03/01/2010 CHS -> DOS FAT/BPB compatibility fix
<1>     mov     al, [esi+bs_FS_MediaAttrib]
<1>     mov     byte [edi+LD_FS_MediaAttrib], al
<1>     ;
<1>     mov     al, [esi+bs_FS_VersionMaj]
<1>     mov     [edi+LD_FS_VersionMajor], al
<1>     ;
<1>     mov     ax, [esi+bs_FS_BytesPerSec]
<1>     mov     [edi+LD_FS_BytesPerSec], ax
<1>     mov     al, [esi+bs_FS_SecPerTrack]
<1>     xor     ah, ah ; 09/12/2017
<1>     mov     [edi+LD_FS_SecPerTrack], ax
<1>     mov     al, [esi+bs_FS_Heads]
<1>     mov     [edi+LD_FS_NumHeads], ax
<1>     ;
<1>     mov     eax, [esi+bs_FS_UnDelDirD]
<1>     mov     [edi+LD_FS_UnDelDirD], eax
<1>     mov     edx, [esi+bs_FS_MATLocation]
<1>     mov     [edi+LD_FS_MATLocation], edx
<1>     mov     eax, [esi+bs_FS_RootDirD]
<1>     mov     [edi+LD_FS_RootDirD], eax
<1>     mov     eax, [esi+bs_FS_BeginSector]
<1>     mov     [edi+LD_FS_BeginSector], eax
<1>     mov     eax, [edi+bs_FS_VolumeSize]
<1>     mov     [edi+LD_FS_VolumeSize], eax
<1>     ;
<1>     mov     eax, edx ; [edi+LD_FS_MATLocation]
<1>     add     eax, [edi+LD_FS_BeginSector]
<1>     mov     esi, edi
<1> mread_hd_fs_MAT_sector:
<1>         ;mov       ebx, DOSBootSectorBuff
<1>     ;mov    ecx, 1
<1>     ; 25/07/2022
<1>     sub     ecx, ecx
<1>     inc     cl
<1>     ; ecx = 1
<1>     call    disk_read
<1>     jc      short loc_validate_hd_fs_partition_retn
<1>     ; EDI will not be changed
<1>     mov     esi, ebx
<1> use_hdfs_mat_sector_params:
<1>     mov     eax, [esi+FS_MAT_DATLocation]
<1>     mov     [edi+LD_FS_DATLocation], eax
<1>     mov     eax, [esi+FS_MAT_DATScount]
<1>     mov     [edi+LD_FS_DATSectors], eax
<1>     mov     eax, [esi+FS_MAT_FreeSectors]
<1>         mov     [edi+LD_FS_FreeSectors], eax
<1>     mov     eax, [esi+FS_MAT_FirstFreeSector]
<1>     mov     [edi+LD_FS_FirstFreeSector], eax
<1>     mov     eax, [edi+LD_FS_RootDirD]
<1>     add     eax, [edi+LD_FS_BeginSector]
<1>     mov     esi, edi
<1> read_hd_fs_RDT_sector:
<1>     mov     ebx, DOSBootSectorBuff
<1>     ;mov    ecx, 1
<1>     mov     cl, 1
<1>     call    disk_read
<1>     jc      short loc_validate_hd_fs_partition_retn
<1>     ; EDI will not be changed
<1>     mov     esi, ebx
<1> use_hdfs_RDT_sector_params:
<1>     mov     eax, [esi+FS_RDT_VolumeSerialNo]
<1>     mov     [edi+LD_FS_VolumeSerial], eax
<1>     push    edi
<1>     ;mov    ecx, 16
<1>     mov     cl, 16
<1>     add     esi, FS_RDT_VolumeName
<1>     add     edi, LD_FS_VolumeName
<1>     rep     movsd ; 64 bytes
<1>     pop     esi
<1>             ; Volume Name Reset
<1>         mov     byte [esi+LD_FS_MediaChanged], 6
<1>     ;
<1>         ;mov       cl, [Last_DOS_DiskNo]
<1>     ;add    cl, 'A'
<1>     ;mov    [esi+LD_FS_Name], cl
<1>
<1> loc_validate_hd_fs_partition_retn:
<1>     retn
<1>
<1> %endif
<1>
<1> load_masterboot:
<1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
<1>     ; 14/07/2020 (Reset function has been removed)
<1>     ;
<1>     ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
<1>     ; 2005 - 2011
<1>     ; input -> DL = drive number
<1> ;   mov     ah, 0Dh ; Alternate disk reset
<1> ;   call    int13h
<1> ;   jnc     short pass_reset_error
<1> ;harddisk_error:
```

```
1036                                    <1> ;   retn
1037                                    <1> ;pass_reset_error:
1038 000073C1 BB[42790100]              <1>     mov     ebx, MasterBootBuff
1039                                    <1>     ;mov    ax, 0201h
1040                                    <1>     ;mov    cx, 1
1041                                    <1>     ; 25/07/2022
1042                                    <1>     ;xor    ecx, ecx
1043                                    <1>     ;inc    cl
1044 000073C6 B101                     <1>     mov     cl, 1
1045                                    <1>     ; ecx = 1
1046 000073C8 89C8                     <1>     mov     eax, ecx ; ch = cylinder = 0
1047 000073CA B402                      <1>     mov     ah, 2  ; chs read
1048                                    <1>     ; eax = 0201h
1049 000073CC 30F6                      <1>     xor     dh, dh ; head = 0
1050 000073CE E838DCFFFF                <1>     call    int13h
1051 000073D3 720C                      <1>     jc      short harddisk_error
1052                                    <1>     ;
1053 000073D5 66813D[407B0100]55-       <1>     cmp     word [MBIDCode], 0AA55h
1053 000073DD AA                        <1>
1054 000073DE 7401                      <1>     je      short load_masterboot_ok
1055 000073E0 F9                        <1>     stc
1056                                    <1> harddisk_error:
1057                                    <1> load_masterboot_ok:
1058 000073E1 C3                        <1>     retn
1059                                    <1>
1060                                    <1> get_free_FAT_sectors:
1061                                    <1>     ; 29/08/2023
1062                                    <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
1063                                    <1>     ; 21/12/2017
1064                                    <1>     ; 29/02/2016
1065                                    <1>     ; 13/02/2016
1066                                    <1>     ; 04/02/2016
1067                                    <1>     ; 07/01/2016 (TRDOS 386 = TRDOS v2.0)
1068                                    <1>     ; 11/07/2010
1069                                    <1>     ; 21/06/2009
1070                                    <1>     ; INPUT: ESI = Logical DOS Drive Description Table address
1071                                    <1>     ; OUTPUT: STC => Error
1072                                    <1>     ;   cf = 0 and EAX = Free FAT sectors
1073                                    <1>     ; Also, related parameters and FAT buffer will be reset and updated
1074                                    <1>
1075 000073E2 31C0                      <1>     xor     eax, eax
1076                                    <1>     ;mov    [esi+LD_FreeSectors], eax ; Reset
1077                                    <1>
1078 000073E4 807E0302                  <1>         cmp     byte [esi+LD_FATType], 2
1079 000073E8 7653                      <1>     jna     short loc_gfc_get_fat_free_clusters
1080                                    <1>
1081                                    <1>     ; 29/02/2016
1082 000073EA 48                        <1>     dec     eax ; 0FFFFFFFFh
1083 000073EB 89463A                    <1>     mov     [esi+LD_BPB+BPB_Reserved], eax ; Free cluster count (reset)
1084 000073EE 89463E                    <1>     mov     [esi+LD_BPB+BPB_Reserved+4], eax ; First Free Cluster (reset)
1085 000073F1 40                        <1>     inc     eax ; 0
1086                                    <1>     ;
1087 000073F2 668B4636                  <1>     mov     ax, [esi+LD_BPB+BPB_FSInfo]
1088 000073F6 03466C                    <1>     add     eax, [esi+LD_StartSector]
1089                                    <1>
1090 000073F9 BB[527C0100]              <1>     mov     ebx, DOSBootSectorBuff
1091                                    <1>     ;mov    ecx, 1
1092                                    <1>     ; 25/07/2022
1093 000073FE 31C9                      <1>     xor     ecx, ecx
1094 00007400 FEC1                      <1>     inc     cl
1095                                    <1>     ; ecx = 1
1096 00007402 E8BBA50000                <1>     call    disk_read
1097 00007407 7301                      <1>     jnc     short loc_gfc_check_fsinfo_signs
1098                                    <1> retn_gfc_get_fsinfo_sec:
1099 00007409 C3                        <1>     retn
1100                                    <1>
1101                                    <1> loc_gfc_check_fsinfo_signs:
1102 0000740A BB[527C0100]              <1>     mov     ebx, DOSBootSectorBuff ; 13/02/2016
1103 0000740F 813B52526141              <1>         cmp     dword [ebx], 41615252h
1104 00007415 7524                      <1>     jne     short retn_gfc_get_fsinfo_stc
1105                                    <1>     ;add    ebx, 484
1106                                    <1>     ;cmp    dword [ebx], 61417272h
1107 00007417 81BBE4010000727241-       <1>     cmp     dword [ebx+484], 61417272h
1107 00007420 61                        <1>
1108 00007421 7518                      <1>     jne     short retn_gfc_get_fsinfo_stc
1109                                    <1>     ;add    ebx, 4
1110                                    <1>     ;mov    eax, [ebx]
1111 00007423 8B83E8010000              <1>     mov     eax, [ebx+488]
1112                                    <1>     ; 29/02/2016
1113 00007429 89463A                    <1>     mov     [esi+LD_BPB+BPB_Reserved], eax ; Free cluster count
1114 0000742C 8B93EC010000              <1>     mov     edx, [ebx+492]
1115                                    <1>     ; 29/08/2023 (BugFix)
1116 00007432 89563E                    <1>     mov     [esi+LD_BPB+BPB_Reserved+4], edx ; First Free Cluster
1117                                    <1>     ; 21/12/2017
1118 00007435 89C3                      <1>     mov     ebx, eax ; (initial value = 0FFFFFFFFh)
1119 00007437 43                        <1>     inc     ebx ; 0FFFFFFFFh -> 0
1120 00007438 7513                      <1>     jnz     short short retn_from_get_free_fat32_clusters
1121 0000743A C3                        <1>     retn
1122                                    <1>
1123                                    <1> retn_gfc_get_fsinfo_stc:
1124 0000743B F9                        <1>     stc
1125 0000743C C3                        <1>     retn
1126                                    <1>
1127                                    <1> loc_gfc_get_fat_free_clusters:
1128                                    <1>     ;mov    eax, 2
1129 0000743D B002                      <1>     mov     al, 2
1130                                    <1>     ;mov    [FAT_CurrentCluster], eax
1131                                    <1> loc_gfc_loop_get_next_cluster:
1132 0000743F E8354B0000                <1>     call    get_next_cluster
1133 00007444 730E                      <1>     jnc     short loc_gfc_free_fat_clusters_cont
1134 00007446 21C0                      <1>     and     eax, eax
1135 00007448 7411                      <1>     jz      short loc_gfc_pass_inc_free_cluster_count
1136                                    <1>
1137                                    <1> retn_from_get_free_fat_clusters:
1138 0000744A 8B4674                    <1>     mov     eax, [esi+LD_FreeSectors] ; Free clusters !
1139                                    <1> retn_from_get_free_fat32_clusters:
1140 0000744D 0FB65E13                  <1>         movzx       ebx, byte [esi+LD_BPB+BPB_SecPerClust]
1141 00007451 F7E3                      <1>         mul     ebx
1142                                    <1>     ;mov    [esi+LD_FreeSectors], eax ; Free sectors
1143                                    <1> retn_get_free_sectors_calc:
1144 00007453 C3                        <1>     retn
1145                                    <1>
1146                                    <1> loc_gfc_free_fat_clusters_cont:
1147 00007454 09C0                      <1>     or      eax, eax
1148 00007456 7503                      <1>     jnz     short loc_gfc_pass_inc_free_cluster_count
1149 00007458 FF4674                    <1>     inc     dword [esi+LD_FreeSectors] ; Free clusters !
1150                                    <1>
1151                                    <1> loc_gfc_pass_inc_free_cluster_count:
1152                                    <1>     ;mov    eax, [FAT_CurrentCluster]
1153 0000745B 89C8                      <1>     mov     eax, ecx ; [FAT_CurrentCluster]
1154 0000745D 3B4678                    <1>     cmp     eax, [esi+LD_Clusters]
1155 00007460 77E8                      <1>     ja      short retn_from_get_free_fat_clusters
1156 00007462 40                        <1>     inc     eax
1157                                    <1>     ;mov    [FAT_CurrentCluster], eax
```

```
00007463 EBDA                <1>     jmp     short loc_gfc_loop_get_next_cluster
                             <1>
                             <1> floppy_drv_init:
                             <1>     ; 19/12/2025 (TRDOS 386 Kenrel v2.0.10)
                             <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
                             <1>     ; 09/12/2017
                             <1>     ; 06/07/2016
                             <1>     ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
                             <1>     ; 24/07/2011
                             <1>     ; 04/07/2009
                             <1>     ; INPUT ->
                             <1>     ;       DL = Drive number (0,1)
                             <1>     ; OUTPUT ->
                             <1>     ;       BL = drive name
                             <1>     ;       BH = drive number
                             <1>     ;       ESI = Logical DOS drv description table
                             <1>     ;       EAX = Volume serial number
                             <1>
00007465 BE[7C660000]        <1>     mov     esi, fd0_type ; 10/01/2016
0000746A BF00010900          <1>     mov     edi, Logical_DOSDisks
0000746F 08D2                <1>     or      dl, dl
00007471 7407                <1>     jz      short loc_drv_init_fd0_fd1
00007473 81C700010000        <1>     add     edi, 100h
00007479 46                  <1>     inc     esi ; fd1_type ; 10/01/2016
                             <1> loc_drv_init_fd0_fd1:
0000747A C6477E00            <1>     mov     byte [edi+LD_MediaChanged], 0
0000747E 803E01              <1>     cmp     byte [esi], 1 ; type (>0 if it is existing)
                             <1>             ; 4 = 1.44 MB, 80 track, 3 1/2"
00007481 7221                <1>     jb      short read_fd_boot_sector_retn
00007483 885702              <1>     mov     [edi+LD_PhyDrvNo], dl
                             <1> read_fd_boot_sector:
00007486 30F6                <1>     xor     dh, dh
00007488 B904000000          <1>     mov     ecx, 4 ; Retry Count
                             <1> read_fd_boot_sector_again:
0000748D 51                  <1>     push    ecx
                             <1>     ;mov    cx, 1
0000748E B101                <1>     mov     cl, 1
00007490 66B80102            <1>     mov     ax, 0201h ; Read 1 sector
00007494 BB[527C0100]        <1>     mov     ebx, DOSBootSectorBuff
00007499 E86DDBFFFF          <1>     call    int13h
0000749E 59                  <1>     pop     ecx
0000749F 7304                <1>     jnc     short use_fd_boot_sector_params
000074A1 E2EA                <1>     loop    read_fd_boot_sector_again
                             <1>
                             <1> read_fd_boot_sector_stc_retn:
000074A3 F9                  <1>     stc
                             <1> read_fd_boot_sector_retn:
000074A4 C3                  <1>     retn
                             <1>
                             <1> use_fd_boot_sector_params:
                             <1>     ;mov    esi, DOSBootSectorBuff
000074A5 89DE                <1>     mov     esi, ebx
000074A7 6681BEFE01000055AA  <1>     cmp     word [esi+BS_Validation], 0AA55h
000074B0 75F1                <1>     jne     short read_fd_boot_sector_stc_retn
                             <1>
                             <1> ; 19/12/2025
                             <1> %if 0
                             <1>         cmp     word [esi+bs_FS_Identifier], 'SF'
                             <1>     ;jne    use_fd_fatfs_boot_sector_params
                             <1>     ; 25/07/2022
                             <1>     je      short use_fdfs_boot_sector_params
                             <1>     jmp     use_fd_fatfs_boot_sector_params
                             <1> use_fdfs_boot_sector_params:
                             <1>     mov     al, [esi+bs_FS_LBA_Ready]
                             <1>     mov     [edi+LD_FS_LBAYes], al
                             <1>     ;
                             <1>     ; 03/01/2010 CHS -> DOS FAT/BPB compatibility fix
                             <1>     mov     al, [esi+bs_FS_MediaAttrib]
                             <1>     mov     [edi+LD_FS_MediaAttrib], al
                             <1>     ;
                             <1>         mov al, [esi+bs_FS_VersionMaj]
                             <1>     mov     byte [edi+LD_FS_VersionMajor], al
                             <1>     mov     ax, [esi+bs_FS_BytesPerSec]
                             <1>     mov     [edi+LD_FS_BytesPerSec], ax
                             <1>     mov     al, [esi+bs_FS_SecPerTrack]
                             <1>     sub     ah, ah ; 09/12/2017
                             <1>     mov     [edi+LD_FS_SecPerTrack], ax
                             <1>     mov     al, [esi+bs_FS_Heads]
                             <1>     mov     [edi+LD_FS_NumHeads], ax
                             <1>     ;
                             <1>     mov     eax, [esi+bs_FS_UnDelDirD]
                             <1>     mov     [edi+LD_FS_UnDelDirD], eax
                             <1>     mov     eax, [esi+bs_FS_MATLocation]
                             <1>     mov     [edi+LD_FS_MATLocation], eax
                             <1>     mov     eax, [esi+bs_FS_RootDirD]
                             <1>     mov     [edi+LD_FS_RootDirD], eax
                             <1>     mov     eax, [esi+bs_FS_BeginSector]
                             <1>     mov     [edi+LD_FS_BeginSector], eax
                             <1>     mov     eax, [esi+bs_FS_VolumeSize]
                             <1>     mov     [edi+LD_FS_VolumeSize], eax
                             <1>     ;
                             <1>     mov     esi, edi
                             <1>     mov     eax, [esi+LD_FS_MATLocation]
                             <1>     ;add    eax, [edi+LD_FS_BeginSector]
                             <1> read_fd_MAT_sector_again:
                             <1>     ;mov    ebx, DOSBootSectorBuff
                             <1>     ;mov    ecx, 1
                             <1>     mov     cl, 1
                             <1>     call    chs_read
                             <1>     mov     esi, ebx
                             <1>     ;jnc    short use_fdfs_mat_sector_params
                             <1>     ;jmp    short read_fd_boot_sector_retn
                             <1>     ;retn
                             <1>     ; 25/07/2022
                             <1>     jc      short read_fd_RDT_sector_retn
                             <1> use_fdfs_mat_sector_params:
                             <1>     mov     eax, [esi+FS_MAT_DATLocation]
                             <1>     mov     [edi+LD_FS_DATLocation], eax
                             <1>     mov     eax, [esi+FS_MAT_DATScount]
                             <1>     mov     [edi+LD_FS_DATSectors], eax
                             <1>     mov     eax, [edi+FS_MAT_FreeSectors]
                             <1>     mov     [edi+LD_FS_FreeSectors], eax
                             <1>     mov     eax, [esi+FS_MAT_FirstFreeSector]
                             <1>     mov     [edi+LD_FS_FirstFreeSector], eax
                             <1>     ;
                             <1>     mov     esi, edi
                             <1>     mov     eax, [esi+LD_FS_RootDirD]
                             <1> read_fd_RDT_sector_again:
                             <1>     ;mov    ebx, DOSBootSectorBuff
                             <1>     ;mov    cx, 1
                             <1>     mov     cl, 1
                             <1>     call    chs_read
                             <1>     mov     esi, ebx
                             <1>     jc      short read_fd_RDT_sector_retn
```

```
1282                            <1> use_fdfs_RDT_sector_params:
1283                            <1>     mov     eax, [esi+FS_RDT_VolumeSerialNo]
1284                            <1>     mov     [edi+LD_FS_VolumeSerial], eax
1285                            <1>     push    edi
1286                            <1>     ;mov    ecx, 16
1287                            <1>     mov     cl, 16
1288                            <1>     add     esi, FS_RDT_VolumeName
1289                            <1>     add     edi, LD_FS_VolumeName
1290                            <1>     rep     movsd ; 64 bytes
1291                            <1>     pop     esi
1292                            <1>     mov     byte [esi+LD_FATType], 0
1293                            <1>     mov     byte [esi+LD_FSType], 0A1h
1294                            <1>         jmp     loc_cont_use_fd_boot_sector_params
1295                            <1>
1296                            <1> read_fd_RDT_sector_stc_retn:
1297                            <1>     stc
1298                            <1> read_fd_RDT_sector_retn:
1299                            <1>     retn
1300                            <1> %endif
1301                            <1>
1302                            <1> use_fd_fatfs_boot_sector_params:
1303 000074B2 807E2629          <1>     cmp     byte [esi+BS_BootSig], 29h
1304                            <1>     ;jne    short read_fd_RDT_sector_stc_retn
1305                            <1>     ; 19/12/2025
1306 000074B6 75EB              <1>     jne     short read_fd_boot_sector_stc_retn
1307 000074B8 807E15F0          <1>     cmp     byte [esi+BPB_Media], 0F0h
1308                            <1>     ;jb     short read_fd_RDT_sector_retn
1309                            <1>     ; 19/12/2025
1310 000074BC 72E6              <1>     jb      short read_fd_boot_sector_retn
1311 000074BE 57                <1>     push    edi
1312 000074BF 83C706            <1>     add     edi, LD_BPB
1313                            <1>     ;mov    ecx, 16
1314 000074C2 B110              <1>     mov     cl, 16
1315 000074C4 F3A5              <1>     rep     movsd ; 64 bytes
1316 000074C6 5E                <1>     pop     esi
1317 000074C7 31C0              <1>     xor     eax, eax
1318 000074C9 89466C            <1>     mov     [esi+LD_StartSector], eax ; 0
1319 000074CC 668B461C          <1>     mov     ax, [esi+LD_BPB+BPB_FATSz16]
1320 000074D0 8A4E16            <1>     mov     cl, [esi+LD_BPB+BPB_NumFATs]
1321 000074D3 F7E1              <1>     mul     ecx
1322                            <1>     ; edx = 0 !
1323 000074D5 668B5614          <1>     mov     dx, [esi+LD_BPB+BPB_RsvdSecCnt]
1324 000074D9 66895660          <1>     mov     [esi+LD_FATBegin], dx
1325                            <1>     ; 25/07/2022
1326 000074DD 01D0              <1>     add     eax, edx
1327                            <1>     ;add    ax, dx
1328 000074DF 894664            <1>     mov     [esi+LD_ROOTBegin], eax
1329 000074E2 894668            <1>     mov     [esi+LD_DATABegin], eax
1330 000074E5 668B5617          <1>     mov     dx, [esi+LD_BPB+BPB_RootEntCnt]
1331                            <1>     ;;shl   edx, 5 ; * 32 (Size of a directory entry)
1332                            <1>     ;shl    dx, 5
1333                            <1>     ;;add   edx, 511
1334                            <1>     ;add    dx, 511
1335                            <1>     ;;shr   edx, 9 ; edx = ((edx*32)+511) / 512
1336                            <1>     ;shr    dx, 9
1337 000074E9 6683C20F          <1>     add     dx, 15 ; 06/07/2016 (+(512/32)-1)
1338                            <1>     ;shr    dx, 4 ; / 16 (==16 entries per sector)
1339                            <1>     ; 25/07/2022
1340 000074ED C1EA04            <1>     shr     edx, 4
1341 000074F0 015668            <1>     add     [esi+LD_DATABegin], edx ; + rd sectors
1342                            <1>     ;movzx  eax, word [esi+LD_BPB+BPB_TotalSec16]
1343 000074F3 668B4619          <1>     mov     ax, [esi+LD_BPB+BPB_TotalSec16]
1344 000074F7 894670            <1>     mov     [esi+LD_TotalSectors], eax
1345 000074FA 2B4668            <1>     sub     eax, [esi+LD_DATABegin]
1346                            <1>     ;movzx  ecx, byte [esi+LD_BPB+BPB_SecPerClust]
1347 000074FD 8A4E13            <1>     mov     cl, [esi+LD_BPB+BPB_SecPerClust]
1348 00007500 80F901            <1>     cmp     cl, 1
1349 00007503 7604              <1>     jna     short save_fd_fatfs_cluster_count
1350                            <1>     ; 25/07/2022
1351 00007505 29D2              <1>     sub     edx, edx
1352                            <1>     ;sub    dx, dx ; 0
1353                            <1>     ;sub    dl, dl ; 06/07/2016
1354 00007507 F7F1              <1>     div     ecx
1355                            <1> save_fd_fatfs_cluster_count:
1356 00007509 894678            <1>     mov     [esi+LD_Clusters], eax
1357                            <1>
1358                            <1>       ; Maximum Valid Cluster Number = EAX +1
1359                            <1>       ; with 2 reserved clusters = EAX +2
1360                            <1>
1361                            <1> reset_FAT_buffer_decriptors:
1362 0000750C 29C0              <1>     sub     eax, eax ; 0
1363 0000750E A2[567E0100]      <1>     mov     [FAT_BuffValidData], al ; 0
1364 00007513 A2[577E0100]      <1>     mov     [FAT_BuffDrvName], al ; 0
1365 00007518 A3[5A7E0100]      <1>     mov     [FAT_BuffSector], eax ; 0
1366                            <1>
1367                            <1> read_fd_FAT_sectors:
1368 0000751D BB001C0900        <1>     mov     ebx, FAT_Buffer
1369 00007522 668B4614          <1>     mov     ax, [esi+LD_BPB+BPB_RsvdSecCnt]
1370                            <1>     ;mov    ecx, 3
1371 00007526 B103              <1>     mov     cl, 3 ; 3 sectors
1372 00007528 E89BA40000        <1>     call    chs_read
1373 0000752D 7240              <1>     jc      short read_fd_FAT_sectors_retn
1374                            <1> use_fd_FAT_sectors:
1375 0000752F 8A4602            <1>     mov     al, [esi+LD_PhyDrvNo]
1376 00007532 0441              <1>     add     al, 'A'
1377 00007534 A2[577E0100]      <1>     mov     [FAT_BuffDrvName], al
1378 00007539 C605[567E0100]01  <1>     mov     byte [FAT_BuffValidData], 1
1379 00007540 E82B000000        <1>     call    fd_init_calculate_free_clusters
1380 00007545 7228              <1>     jc      short read_fd_FAT_sectors_retn
1381                            <1>
1382                            <1> loc_use_fd_boot_sector_params_FAT:
1383 00007547 C6460301          <1>     mov     byte [esi+LD_FATType], 1 ; FAT 12
1384 0000754B C6460401          <1>     mov     byte [esi+LD_FSType], 1
1385 0000754F 8B462D            <1>         mov     eax, [esi+LD_BPB+VolumeID]
1386                            <1> loc_cont_use_fd_boot_sector_params:
1387 00007552 8A7E02            <1>     mov     bh, [esi+LD_PhyDrvNo]
1388 00007555 887E7D            <1>     mov     [esi+LD_DParamEntry], bh
1389 00007558 88FB              <1>     mov     bl, bh
1390 0000755A 80C341            <1>     add     bl, 'A'
1391 0000755D 881E              <1>     mov     byte [esi+LD_Name], bl
1392 0000755F C6460101          <1>     mov     byte [esi+LD_DiskType], 1
1393 00007563 C6460500          <1>     mov     byte [esi+LD_LBAYes], 0
1394 00007567 C6467C00          <1>     mov     byte [esi+LD_PartitionEntry], 0
1395 0000756B C6467E06          <1>     mov     byte [esi+LD_MediaChanged], 6 ; Volume Name Reset
1396                            <1>
1397                            <1> read_fd_FAT_sectors_retn:
1398 0000756F C3                <1>     retn
1399                            <1>
1400                            <1> fd_init_calculate_free_clusters:
1401                            <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
1402                            <1>     ; 09/12/2017
1403                            <1>     ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
1404                            <1>     ; 04/07/2009
1405                            <1>     ; INPUT ->
```

```
1406                                <1>     ;     ESI = Logical DOS drive description table address
1407                                <1>     ; OUTPUT ->
1408                                <1>     ;    [ESI+LD_FreeSectors] will be set
1409                                <1>
1410 00007570 29C0                 <1>     sub     eax, eax
1411 00007572 894674               <1>     mov     [esi+LD_FreeSectors], eax ; 0
1412 00007575 B002                 <1>     mov     al, 2 ; eax = 2
1413                                <1>
1414                                <1> fd_init_loop_get_next_cluster:
1415 00007577 E828000000           <1>     call    fd_init_get_next_cluster
1416 0000757C 7225                 <1>     jc      short fd_init_calculate_free_clusters_retn
1417                                <1>
1418                                <1> fd_init_free_fat_clusters:
1419                                <1>     ;cmp    eax, 0
1420                                <1>     ;ja     short fd_init_pass_inc_free_cluster_count
1421                                <1>     ;and    eax, eax
1422                                <1>     ;jnz    short fd_init_pass_inc_free_cluster_count
1423                                <1>     ;and    ax, ax
1424 0000757E 21C0                 <1>     and     eax, eax ; 25/07/2022
1425 00007580 7503                 <1>     jnz     short fd_init_pass_inc_free_cluster_count
1426                                <1>     ; 25/07/2022
1427 00007582 FF4674               <1>     inc     dword [esi+LD_FreeSectors]
1428                                <1>         ;inc       word [esi+LD_FreeSectors]
1429                                <1>
1430                                <1> fd_init_pass_inc_free_cluster_count:
1431                                <1>     ; 25/07/2022
1432 00007585 A1[527E0100]         <1>     mov     eax, [FAT_CurrentCluster]
1433                                <1>     ;mov    ax, [FAT_CurrentCluster]
1434 0000758A 3B4678               <1>     cmp     eax, [esi+LD_Clusters]
1435                                <1>     ;cmp    ax, [esi+LD_Clusters]
1436 0000758D 7703                 <1>     ja      short short retn_from_fd_init_calculate_free_clusters
1437 0000758F 40                   <1>     inc     eax
1438                                <1>     ;inc    ax
1439 00007590 EBE5                 <1>     jmp     short fd_init_loop_get_next_cluster
1440                                <1>
1441                                <1> retn_from_fd_init_calculate_free_clusters:
1442                                <1>     ; 25/07/2022
1443                                <1>     ;xor    eax, eax
1444 00007592 30E4                 <1>     xor     ah, ah
1445 00007594 8A4613               <1>     mov     al, [esi+LD_BPB+BPB_SecPerClust]
1446 00007597 3C01                 <1>     cmp     al, 1
1447 00007599 7608                 <1>     jna     short fd_init_calculate_free_clusters_retn
1448                                <1>     ;;movzx eax, al
1449                                <1>     ;xor    ah, ah ; 09/12/2017
1450                                <1>     ; 25/07/2022
1451 0000759B 8B4E74               <1>     mov     ecx, [esi+LD_FreeSectors]
1452                                <1>     ;mov    cx, [esi+LD_FreeSectors] ; Count of free clusters
1453 0000759E F7E1                 <1>     mul     ecx
1454                                <1>     ;mul    cx
1455 000075A0 894674               <1>     mov     [esi+LD_FreeSectors], eax
1456                                <1>     ;mov    [esi+LD_FreeSectors], ax
1457                                <1> fd_init_calculate_free_clusters_retn:
1458 000075A3 C3                   <1>     retn
1459                                <1>
1460                                <1> fd_init_get_next_cluster:
1461                                <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
1462                                <1>     ; 04/02/2016
1463                                <1>     ; 02/02/2016
1464                                <1>     ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
1465                                <1>     ; 04/07/2009
1466                                <1>     ; INPUT ->
1467                                <1>     ;    EAX = Current cluster
1468                                <1>     ;    ESI = Logical DOS drive description table address
1469                                <1>     ;    EDX = 0
1470                                <1>     ; OUTPUT ->
1471                                <1>     ;    EAX = Next cluster
1472                                <1>
1473 000075A4 A3[527E0100]         <1>     mov     [FAT_CurrentCluster], eax
1474                                <1> fd_init_get_next_cluster_readnext:
1475 000075A9 29D2                 <1>     sub     edx, edx ; 0
1476 000075AB BB00040000           <1>     mov     ebx, 1024 ; 400h
1477 000075B0 F7F3                 <1>     div     ebx
1478                                <1>     ; EAX = Count of 3 FAT sectors
1479                                <1>     ; EDX = Buffer entry index
1480 000075B2 89C1                 <1>     mov     ecx, eax
1481                                <1>     ;mov    eax, 3
1482 000075B4 B003                 <1>     mov     al, 3
1483 000075B6 F7E2                 <1>     mul     edx ; Multiply by 3
1484                                <1>     ;shr    ax, 1 ; Divide by 2
1485                                <1>     ; 25/07/2022
1486 000075B8 D1E8                 <1>     shr     eax, 1
1487 000075BA 89C3                 <1>     mov     ebx, eax ; Buffer byte offset
1488 000075BC 81C3001C0900         <1>     add     ebx, FAT_Buffer
1489 000075C2 89C8                 <1>     mov     eax, ecx
1490                                <1>     ;mov    edx, 3
1491 000075C4 66BA0300             <1>     mov     dx, 3
1492 000075C8 F7E2                 <1>     mul     edx
1493                                <1>     ; EAX = FAT Beginning Sector
1494                                <1>     ; EDX = 0
1495 000075CA 8A0E                 <1>     mov     cl, [esi+LD_Name]
1496                                <1>     ;cmp    byte [FAT_BuffValidData], 0
1497                                <1>     ;jna    short fd_init_load_FAT_sectors0
1498 000075CC 3A0D[577E0100]       <1>     cmp     cl, [FAT_BuffDrvName]
1499 000075D2 751D                 <1>     jne     short fd_init_load_FAT_sectors0
1500 000075D4 3B05[5A7E0100]       <1>     cmp     eax, [FAT_BuffSector]
1501 000075DA 751B                 <1>     jne     short fd_init_load_FAT_sectors1
1502                                <1>     ; 25/07/2022
1503 000075DC A1[527E0100]         <1>     mov     eax, [FAT_CurrentCluster]
1504                                <1>     ;mov    al, [FAT_CurrentCluster]
1505 000075E1 D1E8                 <1>     shr     eax, 1
1506                                <1>     ;shr    al, 1
1507 000075E3 668B03               <1>     mov     ax, [ebx]
1508 000075E6 7305                 <1>     jnc     short fd_init_gnc_even
1509                                <1>     ;shr    ax, 4
1510                                <1>     ; 25/04/2022
1511 000075E8 C1E804               <1>     shr     eax, 4
1512                                <1> fd_init_gnc_clc_retn:
1513 000075EB F8                   <1>     clc
1514 000075EC C3                   <1>     retn
1515                                <1>
1516                                <1> fd_init_gnc_even:
1517 000075ED 80E40F               <1>     and     ah, 0Fh
1518 000075F0 C3                   <1>     retn
1519                                <1>
1520                                <1> fd_init_load_FAT_sectors0:
1521 000075F1 880D[577E0100]       <1>     mov     [FAT_BuffDrvName], cl
1522                                <1> fd_init_load_FAT_sectors1:
1523 000075F7 C605[567E0100]00     <1>     mov     byte [FAT_BuffValidData], 0
1524 000075FE A3[5A7E0100]         <1>     mov     [FAT_BuffSector], eax
1525 00007603 034660               <1>     add     eax, [esi+LD_FATBegin]
1526 00007606 BB001C0900           <1>     mov     ebx, FAT_Buffer
1527                                <1>     ;movzx  ecx, word [esi+LD_BPB+BPB_FATSz16]
1528 0000760B 668B4E1C             <1>     mov     cx, [esi+LD_BPB+BPB_FATSz16]
1529                                <1>     ;sub    cx, [FAT_BuffSector]
```

```
                                <1>         ; 25/07/2022
0000760F 2B0D[5A7E0100]         <1>     sub     ecx, [FAT_BuffSector]
                                <1>     ;sub    edx, edx
00007615 B203                   <1>     mov     dl, 3
                                <1>     ; edx = 3
                                <1>     ;;cmp   ecx, 3
                                <1>     ;cmp    cx, 3
00007617 39D1                   <1>     cmp     ecx, edx ; 3
00007619 7602                   <1>     jna     short fdinit_pass_fix_sector_count_3
                                <1>     ;;mov   ecx, 3
                                <1>     ;mov    ecx, 3
0000761B 89D1                   <1>     mov     ecx, edx ; 3
                                <1> fdinit_pass_fix_sector_count_3:
0000761D E8A6A30000             <1>     call    chs_read
00007622 730D                   <1>     jnc     short fd_init_FAT_sectors_no_load_error
00007624 C605[567E0100]00       <1>     mov     byte [FAT_BuffValidData], 0
                                <1>             ; Drv not ready or read Error !
0000762B B80F000000             <1>     mov     eax, ERR_DRV_NOT_RDY ; 15
                                <1>     ;xor    edx, edx
00007630 C3                     <1>     retn
                                <1>
                                <1> fd_init_FAT_sectors_no_load_error:
00007631 C605[567E0100]01       <1>     mov     byte [FAT_BuffValidData], 1
00007638 A1[527E0100]           <1>     mov     eax, [FAT_CurrentCluster]
0000763D E967FFFFFF             <1>         jmp     fd_init_get_next_cluster_readnext
                                <1>
                                <1> get_FAT_volume_name:
                                <1>     ; 19/12/2025 (TRDOS 386 Kernel v2.0.10)
                                <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <1>     ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>     ; 12/09/2009
                                <1>     ; INPUT ->
                                <1>     ;       BH = Logical DOS drive number (0,1,2,3,4 ...)
                                <1>     ;        BL = 0
                                <1>     ; OUTPUT ->
                                <1>     ;       CF = 0 -> ESI = Volume name address
                                <1>     ;       CF = 1 -> Root volume name not found
                                <1>
                                <1>     ;mov    ah, 0FFh
                                <1>     ;mov    al, [Last_Dos_DiskNo]
                                <1>     ;cmp    al, bh
                                <1>     ;jb     short loc_gfvn_dir_load_err
                                <1>
00007642 89DE                   <1>     mov     esi, ebx
00007644 81E600FF0000           <1>     and     esi, 0FF00h ; esi = bh
0000764A 81C600010900           <1>     add     esi, Logical_DOSDisks
00007650 8A06                   <1>     mov     al, [esi+LD_Name]
00007652 8A6603                 <1>     mov     ah, [esi+LD_FATType]
                                <1>     ; 19/12/2025
                                <1>     ;cmp    ah, 1
                                <1>     ;jb     short loc_gfvn_dir_load_err
00007655 3C41                   <1>     cmp     al, 'A'
00007657 720C                   <1>     jb      short loc_gfvn_dir_load_err
00007659 80FC02                 <1>     cmp     ah, 2
0000765C 7708                   <1>     ja      short get_FAT32_root_cluster
                                <1>
0000765E E8544A0000             <1>     call    load_FAT_root_directory
00007663 730B                   <1>     jnc     short loc_get_volume_name
                                <1>
                                <1> loc_gfvn_dir_load_err:
                                <1> loc_get_volume_name_retn: ; 29/08/2023
00007665 C3                     <1>     retn
                                <1>
                                <1> get_FAT32_root_cluster:
00007666 8B4632                 <1>     mov     eax, [esi+LD_BPB+BPB_RootClus]
00007669 E8C74A0000             <1>     call    load_FAT_sub_directory
0000766E 72F5                   <1>     jc      short loc_get_volume_name_retn
                                <1>
                                <1> loc_get_volume_name:
00007670 BE00000800             <1>         mov     esi, Directory_Buffer
                                <1>     ;xor    cx, cx ; 0
                                <1>     ; 25/07/2022
00007675 31C9                   <1>     xor     ecx, ecx ; 0
                                <1> check_root_volume_name:
00007677 8A06                   <1>     mov     al, [esi]
00007679 08C0                   <1>     or      al, al
0000767B 74E8                   <1>     jz      short loc_get_volume_name_retn
0000767D 807E0B08               <1>     cmp     byte [esi+0Bh], 08h
00007681 74E2                   <1>     je      short loc_get_volume_name_retn
00007683 663B0D[6C7E0100]       <1>     cmp     cx, [DirBuff_LastEntry]
0000768A 7306                   <1>     jnb     short pass_check_root_volume_name
                                <1>     ;inc    cx
                                <1>     ; 25/07/2022
0000768C 41                     <1>     inc     ecx
0000768D 83C620                 <1>     add     esi, 32
00007690 EBE5                   <1>     jmp     short check_root_volume_name
                                <1>
                                <1>     ; 29/08/2023
                                <1> ;loc_get_volume_name_retn:
                                <1>     ;retn
                                <1>
                                <1> pass_check_root_volume_name:
00007692 803D[687E0100]03       <1>     cmp     byte [DirBuff_FATType], 3
00007699 7230                   <1>     jb      short loc_get_volume_name_retn_xor
                                <1>
0000769B BB001C0900             <1>     mov     ebx, FAT_Buffer
000076A0 BE00010900             <1>     mov     esi, Logical_DOSDisks
000076A5 31C0                   <1>     xor     eax, eax
000076A7 8A25[677E0100]         <1>     mov     ah, [DirBuff_DRV]
000076AD 80EC41                 <1>     sub     ah, 'A'
000076B0 01C6                   <1>     add     esi, eax
000076B2 A1[6E7E0100]           <1>     mov     eax, [DirBuff_Cluster]
000076B7 E8BD480000             <1>     call    get_next_cluster
000076BC 7305                   <1>     jnc     short loc_gfvn_load_FAT32_dir_cluster
                                <1>
000076BE 83F801                 <1>     cmp     eax, 1
000076C1 F5                     <1>     cmc
000076C2 C3                     <1>     retn
                                <1>
                                <1> loc_gfvn_load_FAT32_dir_cluster:
000076C3 E86D4A0000             <1>     call    load_FAT_sub_directory
000076C8 73A6                   <1>     jnc     short loc_get_volume_name
000076CA C3                     <1>     retn
                                <1>
                                <1> loc_get_volume_name_retn_xor:
000076CB 31C0                   <1>     xor     eax, eax
000076CD C3                     <1>     retn
                                <1>
                                <1> get_media_change_status:
                                <1>     ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>     ; 09/09/2009
                                <1>     ; INPUT:
                                <1>     ;     DL = Drive number (physical)
                                <1>     ; OUTPUT: clc & AH = 6 media changed
```

```
1654                                    <1>     ;     clc & AH = 0 media not changed
1655                                    <1>     ;     stc -> Drive not ready or an error
1656                                    <1>
1657 000076CE B416                     <1>     mov     ah, 16h
1658 000076D0 E836D9FFFF               <1>     call    int13h
1659 000076D5 80FC06                   <1>     cmp     ah, 06h
1660 000076D8 7405                     <1>     je      short loc_gmc_status_retn
1661 000076DA 08E4                     <1>     or      ah, ah
1662 000076DC 7401                     <1>     jz      short loc_gmc_status_retn
1663                                    <1> loc_gmc_status_stc_retn:
1664 000076DE F9                       <1>     stc
1665                                    <1> loc_gmc_status_retn:
1666 000076DF C3                       <1>     retn
3434                                        %include 'trdosk3.s' ; 06/01/2016
   1                                    <1> ; ****************************************************************************
   2                                    <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.10) - MAIN PROGRAM : trdosk3.s
   3                                    <1> ; ----------------------------------------------------------------------------
   4                                    <1> ; Last Update: 19/12/2025  (Previous: 26/09/2024, TRDOS 386 v2.0.9)
   5                                    <1> ; ----------------------------------------------------------------------------
   6                                    <1> ; Beginning: 06/01/2016
   7                                    <1> ; ----------------------------------------------------------------------------
   8                                    <1> ; Assembler: NASM version 2.15 (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>     ; 19/12/2025 (TRDOS 386 Kernel v2.0.10)
  21                                    <1>     ; 26/07/2022 (TRDOS 386 Kernel v2.0.5)
  22                                    <1>     ; 16/10/2016
  23                                    <1>     ; 02/02/2016
  24                                    <1>     ; 15/01/2016 (TRDOS 386 = TRDOS v2.0)
  25                                    <1>     ; 18/08/2011
  26                                    <1>     ; 09/09/2009
  27                                    <1>     ; INPUT:
  28                                    <1>     ;   DL = Logical DOS Drive Number
  29                                    <1>     ; OUTPUT:
  30                                    <1>     ;  cf=1 -> Not successful
  31                                    <1>     ;   EAX = Error code
  32                                    <1>     ;  cf=0 ->
  33                                    <1>     ;   EAX = 0 (successful)
  34                                    <1>
  35 000076E0 31DB                      <1>     xor     ebx, ebx
  36 000076E2 88D7                      <1>     mov     bh, dl
  37                                    <1>
  38                                    <1>     ;cmp    dl, 1
  39                                    <1>     ;jna    short loc_ccdrv_initial_media_change_check
  40                                    <1>     ;cmp    bh, [Last_Dos_DiskNo]
  41                                    <1>     ;ja     short loc_ccdrv_drive_not_ready_err
  42                                    <1>
  43                                    <1> loc_ccdrv_initial_media_change_check:
  44 000076E4 BE00010900                <1>     mov     esi, Logical_DOSDisks
  45 000076E9 01DE                      <1>     add     esi, ebx
  46                                    <1> loc_ccdrv_dos_drive_name_check:
  47 000076EB 80FA02                    <1>     cmp     dl, 2
  48 000076EE 720E                      <1>     jb      short loc_ccdrv_dos_drive_name_check_ok
  49                                    <1>
  50 000076F0 8A06                      <1>     mov     al, [esi+LD_Name]
  51 000076F2 2C41                      <1>     sub     al, 'A'
  52 000076F4 38D0                      <1>     cmp     al, dl
  53 000076F6 7406                      <1>     je      short loc_ccdrv_dos_drive_name_check_ok
  54                                    <1>
  55                                    <1> loc_ccdrv_drive_not_ready_err:
  56                                    <1>     ; 16/10/2016 (15h -> 15)
  57                                    <1>     ;mov    eax, 15 ; Drive not ready
  58                                    <1>     ; 26/07/2022
  59 000076F8 29C0                      <1>     sub     eax, eax
  60 000076FA B00F                      <1>     mov     al, 15
  61                                    <1> loc_change_current_drive_stc_retn:
  62 000076FC F9                        <1>     stc
  63 000076FD C3                        <1>     retn
  64                                    <1>
  65                                    <1> loc_ccdrv_dos_drive_name_check_ok:
  66 000076FE 8A667E                    <1>     mov     ah, [esi+LD_MediaChanged]
  67 00007701 80FC06                    <1>     cmp     ah, 6  ; VOLUME NAME CHECK/MOVE SIGN
  68 00007704 7455                      <1>     je      short loc_ccdrv_get_FAT_volume_name_0
  69                                    <1>
  70 00007706 80FA01                    <1>     cmp     dl, 1
  71 00007709 7778                      <1>     ja      short loc_gmcs_init_drv_hd
  72                                    <1>
  73                                    <1> loc_gmcs_init_drv_fd:
  74 0000770B 08E4                      <1>     or      ah, ah
  75                                    <1>     ; AH = 1 is initialization sign (invalid_fd_parameter)
  76 0000770D 7517                      <1>     jnz     short loc_ccdrv_call_fd_init
  77                                    <1>
  78 0000770F E8BAFFFFFF                <1>     call    get_media_change_status
  79 00007714 72E2                      <1>     jc      short loc_ccdrv_drive_not_ready_err
  80                                    <1>
  81 00007716 20E4                      <1>     and     ah, ah
  82 00007718 7471                      <1>     jz      short loc_change_current_drv3
  83                                    <1>
  84 0000771A 80F406                    <1>     xor     ah, 6
  85 0000771D 75D9                      <1>     jnz     short loc_ccdrv_drive_not_ready_err
  86                                    <1>
  87                                    <1> loc_ccdrv_call_fd_init_check_vol_id:
  88 0000771F E81C090000                <1>     call    get_volume_serial_number
  89 00007724 730D                      <1>     jnc     short loc_ccdrv_check_vol_serial
  90                                    <1>
  91                                    <1> loc_ccdrv_call_fd_init:
  92 00007726 E83AFDFFFF                <1>     call    floppy_drv_init
  93 0000772B 731A                      <1>     jnc     short loc_reset_drv_fd_current_dir
  94                                    <1>
  95                                    <1> loc_ccdrv_fdinit_fail_retn:
  96                                    <1>     ; 16/10/2016
  97 0000772D B80F000000                <1>     mov     eax, 15 ; Drive not ready
  98 00007732 C3                        <1>     retn
  99                                    <1>
 100                                    <1> loc_ccdrv_check_vol_serial:
 101 00007733 A3[38770100]              <1>     mov     [Current_VolSerial], eax
 102                                    <1>     ;mov    dl, bh
 103 00007738 E828FDFFFF                <1>     call    floppy_drv_init
 104 0000773D 72EE                      <1>     jc      short loc_ccdrv_fdinit_fail_retn
 105                                    <1>
 106 0000773F 3B05[38770100]            <1>     cmp     eax, [Current_VolSerial]
 107 00007745 7440                      <1>     je      short loc_change_current_drv2
 108                                    <1>
 109                                    <1> loc_reset_drv_fd_current_dir:
 110 00007747 31C0                      <1>     xor     eax, eax
```

```
111 00007749 88467F              <1>         mov [esi+LD_CDirLevel], al
112 0000774C 89F7                <1>     mov     edi, esi
113 0000774E 81C780000000        <1>     add     edi, LD_CurrentDirectory
114 00007754 B920000000          <1>     mov     ecx, 32
115 00007759 F3AB                <1>     rep     stosd
116                              <1>
117                              <1> loc_ccdrv_get_FAT_volume_name_0:
118                              <1>     ; 19/12/2025
119                              <1>     ;mov    al, [esi+LD_FATType]
120                              <1>     ;or     al, al
121                              <1>     ;jz     short loc_change_current_drv2
122                              <1>
123 0000775B 56                  <1>     push    esi
124                              <1>     ;cmp    al, 2
125                              <1>     ; 19/12/2025
126 0000775C 807E0302            <1>     cmp     byte [esi+LD_FATType], 2
127 00007760 7705                <1>     ja      short loc_ccdrv_get_FAT32_vol_name
128                              <1>
129                              <1> loc_ccdrv_get_FAT2_16_vol_name:
130 00007762 83C631              <1>     add     esi, LD_BPB + VolumeLabel
131 00007765 EB03                <1>     jmp     short loc_ccdrv_get_FAT_volume_name_1
132                              <1>
133                              <1> loc_ccdrv_get_FAT32_vol_name:
134 00007767 83C64D              <1>     add     esi, LD_BPB + FAT32_VolLab
135                              <1> loc_ccdrv_get_FAT_volume_name_1:
136 0000776A 53                  <1>     push    ebx
137 0000776B 56                  <1>     push    esi
138 0000776C E8D1FEFFFF          <1>     call    get_FAT_volume_name
139 00007771 5F                  <1>     pop     edi
140 00007772 5B                  <1>     pop     ebx
141                              <1>     ; BL = 0
142 00007773 720B                <1>     jc      short loc_change_current_drv1
143 00007775 20C0                <1>     and     al, al
144 00007777 7407                <1>     jz      short loc_change_current_drv1
145                              <1>
146                              <1> loc_ccdrv_move_FAT_volume_name:
147 00007779 B90B000000          <1>     mov     ecx, 11
148 0000777E F3A4                <1>     rep     movsb
149                              <1>
150                              <1> loc_change_current_drv1:
151 00007780 5E                  <1>     pop     esi
152 00007781 EB04                <1>     jmp     short loc_change_current_drv2
153                              <1>
154                              <1> loc_gmcs_init_drv_hd:
155 00007783 08E4                <1>     or      ah, ah
156 00007785 7404                <1>     jz      short loc_change_current_drv3
157                              <1>     ; BL = 0, BH = Logical DOS drive number
158                              <1> loc_change_current_drv2:
159 00007787 C6467E00            <1>     mov     byte [esi+LD_MediaChanged], 0
160                              <1> loc_change_current_drv3:
161 0000778B 883D[42770100]      <1>     mov     [Current_Drv], bh
162                              <1>
163                              <1>     ;call   restore_current_directory
164                              <1>     ;retn
165                              <1>
166                              <1> restore_current_directory:
167                              <1>     ; 19/12/2025 (TRDOS 386 Kernel v2.0.10)
168                              <1>     ; 26/07/2022 (TRDOS 386 Kernel v2.0.5)
169                              <1>     ; 11/02/2016
170                              <1>     ; 15/01/2016 (TRDOS 386 = TRDOS v2.0)
171                              <1>     ; 25/01/2010
172                              <1>     ; 12/10/2009
173                              <1>     ;
174                              <1>     ; INPUT:
175                              <1>     ;   ESI = Logical DOS Drive Description Table
176                              <1>     ;
177                              <1>     ; OUTPUT:
178                              <1>     ;   ESI = Logical DOS Drive Description Table
179                              <1>     ;   EDI = offset Current_Dir_Drv
180                              <1>
181 00007791 8A4603              <1>     mov     al, [esi+LD_FATType]
182 00007794 A2[41770100]        <1>     mov     [Current_FATType], al
183                              <1>
184 00007799 8A26                <1>     mov     ah, [esi+LD_Name]
185 0000779B 8825[43770100]      <1>     mov     [Current_Dir_Drv], ah
186                              <1>
187                              <1>     ; 19/12/2025
188                              <1>     ;and    al, al
189                              <1>     ;jz     short loc_restore_FS_current_directory
190                              <1>
191                              <1> loc_restore_FAT_current_directory:
192 000077A1 8A667F              <1>     mov     ah, [esi+LD_CDirLevel]
193 000077A4 8825[40770100]      <1>     mov     [Current_Dir_Level], ah
194 000077AA 08E4                <1>     or      ah, ah
195 000077AC 7410                <1>         jz  short loc_ccdrv_reset_cdir_FAT_12_16_32_fcluster
196                              <1>
197 000077AE 0FB6D4              <1>     movzx   edx, ah
198 000077B1 C0E204              <1>     shl     dl, 4 ; * 16
199 000077B4 01F2                <1>         add edx, esi
200 000077B6 8B828C000000        <1>     mov     eax, [edx+LD_CurrentDirectory+12]
201 000077BC EB24                <1>     jmp     short loc_ccdrv_reset_cdir_FAT_fcluster
202                              <1>
203                              <1> ; 19/12/2025
204                              <1> %if 0
205                              <1> loc_restore_FS_current_directory:
206                              <1>     ;call   load_current_FS_directory
207                              <1>     ;retn
208                              <1>     ; 26/07/2022
209                              <1>     jmp     load_current_FS_directory
210                              <1> %endif
211                              <1>
212                              <1> loc_ccdrv_reset_cdir_FAT_12_16_32_fcluster:
213 000077BE 3C03                <1>     cmp     al, 3
214 000077C0 7205                <1>     jb      short loc_ccdrv_reset_cdir_FAT_12_16_fcluster
215                              <1> loc_ccdrv_reset_cdir_FAT32_fcluster:
216 000077C2 8B4632              <1>     mov     eax, [esi+LD_BPB+FAT32_RootFClust]
217 000077C5 EB02                <1>     jmp     short loc_ccdrv_check_rootdir_sign
218                              <1> loc_ccdrv_reset_cdir_FAT_12_16_fcluster:
219                              <1>     ;xor    al, al  ; xor eax, eax
220                              <1>     ; 26/07/2022
221 000077C7 31C0                <1>     xor     eax, eax
222                              <1>     ;xor    edx, edx
223                              <1> loc_ccdrv_check_rootdir_sign:
224 000077C9 80BE8000000000      <1>     cmp     byte [esi+LD_CurrentDirectory], 0
225 000077D0 7510                <1>     jne     short loc_ccdrv_reset_cdir_FAT_fcluster
226                              <1> loc_ccdrv_set_rootdir_FAT_fcluster:
227 000077D2 89868C000000        <1>         mov     [esi+LD_CurrentDirectory+12], eax
228 000077D8 C786800000000524F4F- <1>     mov     dword [esi+LD_CurrentDirectory], 'ROOT'
228 000077E1 54                  <1>
229                              <1>
230                              <1> loc_ccdrv_reset_cdir_FAT_fcluster:
231 000077E2 A3[3C770100]        <1>     mov     [Current_Dir_FCluster], eax
232                              <1>
233 000077E7 BF[A07E0100]        <1>     mov     edi, PATH_Array
```

```
234 000077EC 89F2                <1>     mov     edx, esi
235 000077EE 81C680000000        <1>     add     esi, LD_CurrentDirectory
236 000077F4 B920000000          <1>     mov     ecx, 32
237 000077F9 F3A5                <1>     rep     movsd
238                              <1>
239 000077FB E8042B0000          <1>     call    change_prompt_dir_string
240                              <1>
241 00007800 89D6                <1>     mov     esi, edx
242                              <1>
243 00007802 29C0                <1>         sub eax, eax
244                              <1>        ;sub edx, edx
245 00007804 BF[43770100]        <1>     mov     edi, Current_Dir_Drv
246                              <1>
247 00007809 A2[5D2E0100]        <1>     mov     [Restore_CDIR], al ; 0
248 0000780E C3                  <1>     retn
249                              <1>
250                              <1> dos_prompt:
251                              <1>     ; 26/07/2022 (TRDOS 386 Kernel v2.0.5)
252                              <1>     ; 06/05/2016
253                              <1>     ; 30/01/2016
254                              <1>     ; 29/01/2016
255                              <1>     ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
256                              <1>     ; 15/09/2011
257                              <1>     ; 13/09/2009
258                              <1>     ; 2004-2005
259                              <1>
260                              <1>     ; 06/05/2016
261 0000780F C705[00830100]-     <1>     mov     dword [mainprog_return_addr], return_from_cmd_interpreter
261 00007815 [C1780000]          <1>
262                              <1>
263                              <1> loc_TRDOS_prompt:
264 00007819 BF[42780100]        <1>     mov     edi, TextBuffer
265 0000781E C6075B              <1>     mov     byte [edi], "["
266 00007821 47                  <1>     inc     edi
267 00007822 BE[B92E0100]        <1>     mov     esi, TRDOSPromptLabel
268                              <1> get_next_prompt_label_char:
269 00007827 803E20              <1>     cmp     byte [esi], 20h
270 0000782A 7203                <1>     jb      short pass_prompt_label
271 0000782C A4                  <1>     movsb
272 0000782D EBF8                <1>     jmp     short get_next_prompt_label_char
273                              <1> pass_prompt_label:
274 0000782F C6075D              <1>     mov     byte [edi], "]"
275 00007832 47                  <1>     inc     edi
276 00007833 C60720              <1>     mov     byte [edi], 20h
277 00007836 47                  <1>     inc     edi
278 00007837 BE[43770100]        <1>     mov     esi, Current_Dir_Drv
279 0000783C 66A5                <1>     movsw
280 0000783E A4                  <1>     movsb
281                              <1> loc_prompt_current_directory:
282 0000783F 803E20              <1>     cmp     byte [esi], 20h
283 00007842 7203                <1>     jb      short pass_prompt_current_directory
284 00007844 A4                  <1>     movsb
285 00007845 EBF8                <1>     jmp     short loc_prompt_current_directory
286                              <1> pass_prompt_current_directory:
287 00007847 C6073E              <1>     mov     byte [edi], '>'
288 0000784A 47                  <1>     inc     edi
289 0000784B C60700              <1>     mov     byte [edi], 0
290 0000784E BE[42780100]        <1>     mov     esi, TextBuffer
291 00007853 E852F5FFFF          <1>     call    print_msg
292                              <1>
293                              <1>     ;sub    bh, bh ; video page = 0
294                              <1>     ;call   get_cpos ; get cursor position
295 00007858 668B15[9E760100]    <1>     mov     dx, [CURSOR_POSN] ; video page 0
296 0000785F 8815[A2770100]      <1>     mov     [CursorColumn], dl
297                              <1>
298                              <1>     ; 30/01/2016 (to show cursor on the row, again)
299                              <1>     ; (Initial color attributes of video page 0 is 0)
300                              <1>     ; (see: 'StartPMP' in trdos386.s)
301                              <1>     ;
302                              <1>     ;mov    edi, 0B8000h ; start of video page 0
303                              <1>     ;movzx  ecx, dl ; column
304                              <1>     ;mov    al, 80
305                              <1>     ;mul    dh
306                              <1>     ;add    ax, cx
307                              <1>     ;shl    ax, 1 ; character + attribute
308                              <1>     ;add    di, ax ; (2*80*row) + (2*column)
309                              <1>     ;neg    cl
310                              <1>     ;add    cl, 80
311                              <1>     ;mov    ax, 700h ;  ah = 7 (color attribute)
312                              <1>     ;rep    stosw
313                              <1>
314                              <1> loc_rw_char:
315 00007865 E8AB000000          <1>     call    rw_char
316                              <1> loc_move_command:
317 0000786A BE[F2770100]        <1>     mov     esi, CommandBuffer
318 0000786F 89F7                <1>     mov     edi, esi
319 00007871 31C9                <1>     xor     ecx, ecx
320                              <1> first_command_char:
321 00007873 AC                  <1>     lodsb
322 00007874 3C20                <1>     cmp     al, 20h
323 00007876 770C                <1>     ja      short pass_space_control
324 00007878 723F                <1>     jb      short loc_move_cmd_arguments_ok
325 0000787A 81FE[41780100]      <1>     cmp     esi, CommandBuffer + 79
326 00007880 72F1                <1>     jb      short first_command_char
327 00007882 EB35                <1>     jmp     short loc_move_cmd_arguments_ok
328                              <1>
329                              <1>     ; 26/07/2022
330                              <1> pass_space_control:
331 00007884 3C61                <1>     cmp     al, 61h
332 00007886 7206                <1>     jb      short pass_capitalize
333 00007888 3C7A                <1>     cmp     al, 7Ah
334 0000788A 7702                <1>     ja      short pass_capitalize
335 0000788C 24DF                <1>     and     al, 0DFh
336                              <1> pass_capitalize:
337 0000788E AA                  <1>     stosb
338 0000788F FEC1                <1>     inc     cl
339 00007891 81FE[41780100]      <1>     cmp     esi, CommandBuffer + 79
340                              <1>     ;jb     short next_command_char
341                              <1>     ; 26/07/2022
342 00007897 7320                <1>     jnb     short loc_move_cmd_arguments_ok
343                              <1>
344                              <1> next_command_char:
345 00007899 AC                  <1>     lodsb
346 0000789A 3C20                <1>     cmp     al, 20h
347 0000789C 77E6                <1>     ja      short pass_space_control
348 0000789E 7219                <1>     jb      short loc_move_cmd_arguments_ok
349                              <1>
350                              <1> loc_1st_cmd_arg: ; 30/01/2016
351 000078A0 AC                  <1>     lodsb
352 000078A1 3C20                <1>     cmp     al, 20h
353 000078A3 74FB                <1>     je      short loc_1st_cmd_arg
354 000078A5 7212                <1>     jb      short loc_move_cmd_arguments_ok
355                              <1>
356 000078A7 C60700              <1>         mov     byte [edi], 0
```

```
000078AA 47                  <1>     inc     edi
                             <1>
                             <1> loc_move_cmd_arguments:
000078AB AA                  <1>     stosb
000078AC 81FE[41780100]      <1>     cmp     esi, CommandBuffer + 79
000078B2 7305                <1>     jnb     short loc_move_cmd_arguments_ok
000078B4 AC                  <1>         lodsb
000078B5 3C20                <1>     cmp     al, 20h
000078B7 73F2                <1>     jnb     short loc_move_cmd_arguments
                             <1>     ; 26/07/2022
                             <1>     ;jmp    short loc_move_cmd_arguments_ok
                             <1>
                             <1> ; 26/07/2022
                             <1> ;pass_space_control:
                             <1> ;   cmp     al, 61h
                             <1> ;   jb      short pass_capitalize
                             <1> ;   cmp     al, 7Ah
                             <1> ;   ja      short pass_capitalize
                             <1> ;   and     al, 0DFh
                             <1> ;pass_capitalize:
                             <1> ;   stosb
                             <1> ;   inc     cl
                             <1> ;   cmp     esi, CommandBuffer + 79
                             <1> ;   jb      short next_command_char
                             <1>
                             <1> loc_move_cmd_arguments_ok:
000078B9 C60700              <1>         mov     byte [edi], 0
                             <1>
                             <1> call_command_interpreter:
000078BC E8A8070000          <1>     call    command_interpreter
                             <1>
                             <1> return_from_cmd_interpreter:
000078C1 B950000000          <1>         mov ecx, 80
                             <1>     ;mov    cx, 80
000078C6 BF[F2770100]        <1>     mov     edi, CommandBuffer
000078CB 30C0                <1>     xor     al, al
000078CD F3AA                <1>     rep     stosb
                             <1>     ;cmp    byte [Program_Exit], 0
                             <1>     ;ja     short loc_terminate_trdos
                             <1>
                             <1>     ; 16/01/2016
000078CF 803D[1E680000]03    <1>     cmp     byte [CRT_MODE], 3 ; 80*25 color
000078D6 7419                <1>     je      short pass_set_txt_mode
                             <1>
000078D8 E8AAA2FFFF          <1>     call    set_txt_mode ; set vide mode to 03h
                             <1>     ; 07/01/2017
000078DD 30C0                <1>     xor     al, al
                             <1>
                             <1> loc_check_active_page:
                             <1>     ;xor    al, al
000078DF 3805[AE760100]      <1>     cmp     [ACTIVE_PAGE], al ; 0
                             <1>         ;je loc_TRDOS_prompt
                             <1>     ; 26/07/2022
000078E5 7405                <1>     je      short loc_prompt_again
                             <1>
                             <1>     ; AL = 0 = video page 0
000078E7 E8C3A6FFFF          <1>     call    set_active_page
                             <1> loc_prompt_again: ; 26/07/2022
000078EC E928FFFFFF          <1>         jmp     loc_TRDOS_prompt ; infinitive loop
                             <1>
                             <1> pass_set_txt_mode:
000078F1 BE[E7380100]        <1>     mov     esi, nextline
000078F6 E8AFF4FFFF          <1>     call    print_msg
000078FB EBE2                <1>     jmp     short loc_check_active_page
                             <1>
                             <1> ;rw_char:
                             <1>     ; 26/07/2022 (TRDOS 386 Kernel v2.0.5)
                             <1> loc_arrow:
000078FD 80FC4B              <1>     cmp     ah, 4Bh
00007900 7426                <1>     je      short loc_back
00007902 80FC53              <1>     cmp     ah, 53h
00007905 7421                <1>     je      short loc_back
00007907 80FC4D              <1>     cmp     ah, 4Dh
0000790A 7509                <1>     jne     short readnextchar
0000790C 80FA4F              <1>     cmp     dl, 79
0000790F 7304                <1>     jnb     short readnextchar
00007911 FEC2                <1>     inc     dl
00007913 EB1D                <1>     jmp     short set_cursor_pos
                             <1>
                             <1> rw_char:
                             <1>     ; 26/07/2022 (TRDOS 386 Kernel v2.0.5)
                             <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:
00007915 30E4                <1>     xor     ah, ah
00007917 E80096FFFF          <1>     call    int16h
0000791C 20C0                <1>     and     al, al
0000791E 74DD                <1>     jz      short loc_arrow
00007920 3CE0                <1>     cmp     al, 0E0h
00007922 74D9                <1>     je      short loc_arrow
00007924 3C08                <1>     cmp     al, 08h
00007926 752A                <1>     jne     short char_return
                             <1> loc_back:
00007928 3A15[A2770100]      <1>     cmp     dl, [CursorColumn]
0000792E 76E5                <1>     jna     short readnextchar
                             <1> prev_column:
00007930 FECA                <1>     dec     dl
                             <1> set_cursor_pos:
                             <1>     ;push   dx
00007932 52                  <1>     push    edx ; 29/12/2017
                             <1>     ;xor    bh, bh ; 0 = video page 0
                             <1>     ; DH = Row, DL = Column
00007933 E83BAAFFFF          <1>     call    _set_cpos ; 17/01/2016
00007938 5A                  <1>         pop edx ; 29/12/2017
                             <1>     ;pop    dx
                             <1>     ;movzx  ebx, dl
00007939 88D3                <1>     mov     bl, dl
0000793B 2A1D[A2770100]      <1>     sub     bl, [CursorColumn]
00007941 B020                <1>     mov     al, 20h
00007943 8883[F2770100]      <1>     mov     [CommandBuffer+ebx], al
                             <1>     ;sub    bh, bh ; video page 0
                             <1>     ;mov    cx, 1
00007949 B307                <1>     mov     bl, 7 ; color attribute
                             <1>     ; bh = 0 ; 26/07/2022
0000794B E80AA9FFFF          <1>     call    _write_c_current ; 17/01/2016
```

```
                                <1>     ;mov    dx, [CURSOR_POSN]
00007950 EBC3                   <1>     jmp     short readnextchar
                                <1>
                                <1>     ; 26/07/2022
                                <1> ;loc_arrow:
                                <1> ;   cmp     ah, 4Bh
                                <1> ;   je      short loc_back
                                <1> ;   cmp     ah, 53h
                                <1> ;   je      short loc_back
                                <1> ;   cmp     ah, 4Dh
                                <1> ;   jne     short readnextchar
                                <1> ;   cmp     dl, 79
                                <1> ;   jnb     short readnextchar
                                <1> ;   inc     dl
                                <1> ;   jmp     short set_cursor_pos
                                <1>
                                <1> char_return:
00007952 0FB6DA                 <1>     movzx   ebx, dl
00007955 2A1D[A2770100]         <1>     sub     bl, [CursorColumn]
0000795B 3C20                   <1>     cmp     al, 20h
0000795D 721D                   <1>     jb      short loc_escape
0000795F 8883[F2770100]         <1>     mov     [CommandBuffer+ebx], al
00007965 80FA4F                 <1>     cmp     dl, 79
00007968 73AB                   <1>     jnb     short readnextchar
0000796A 66BB0700               <1>     mov     bx, 7 ; color attribute
0000796E E868A9FFFF             <1>     call    _write_tty
00007973 668B15[9E760100]       <1>     mov     dx, [CURSOR_POSN] ; video page 0
0000797A EB99                   <1>         jmp short readnextchar ; 26/07/2022
                                <1>
                                <1> loc_escape:
0000797C 3C1B                   <1>     cmp     al, 1Bh
0000797E 7414                   <1>     je      short rw_char_retn
                                <1>     ;
00007980 3C0D                   <1>     cmp     al, 0Dh ; CR
00007982 7591                   <1>         jne short readnextchar ; 26/07/2022
                                <1>
                                <1>     ; 13/05/2016
00007984 66BB0700               <1>     mov     bx, 7 ; attribute/color (bl)
                                <1>                   ; video page 0 (bh=0)
00007988 E84EA9FFFF             <1>     call    _write_tty
                                <1>     ;mov    bx, 7  ; attribute/color
                                <1>                    ; video page 0 (bh=0)
0000798D B00A                   <1>     mov     al, 0Ah ; LF
0000798F E847A9FFFF             <1>     call    _write_tty
                                <1> rw_char_retn:
00007994 C3                     <1>     retn
                                <1>
                                <1> show_date:
                                <1>     ; 26/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <1>     ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>        ; 2004-2005
                                <1>
                                <1>     ;mov    ah, 04h
                                <1>     ;call   int1Ah
00007995 E822EAFFFF             <1>     call    RTC_40  ; GET RTC DATE
                                <1>
0000799A 88D0                   <1>     mov     al, dl
0000799C E86E95FFFF             <1>     call    bcd_to_ascii
000079A1 66A3[A52F0100]         <1>     mov     [Day], ax
                                <1>
000079A7 88F0                   <1>     mov     al, dh
000079A9 E86195FFFF             <1>     call    bcd_to_ascii
000079AE 66A3[A82F0100]         <1>     mov     [Month], ax
                                <1>
000079B4 88E8                   <1>     mov     al, ch
000079B6 E85495FFFF             <1>     call    bcd_to_ascii
000079BB 66A3[AB2F0100]         <1>     mov     [Century], ax
                                <1>
000079C1 88C8                   <1>     mov     al, cl
000079C3 E84795FFFF             <1>     call    bcd_to_ascii
000079C8 66A3[AD2F0100]         <1>     mov     [Year], ax
                                <1>
000079CE BE[952F0100]           <1>     mov     esi, Msg_Show_Date
                                <1>     ;call   print_msg
                                <1>     ;retn
                                <1>     ; 26/07/2022
000079D3 E9D2F3FFFF             <1>     jmp     print_msg
                                <1>
                                <1> set_date:
                                <1>     ; 26/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <1>     ; 13/05/2016
                                <1>     ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>        ; 2004-2005
                                <1>
000079D8 BE[792F0100]           <1>     mov     esi, Msg_Enter_Date
000079DD E8C8F3FFFF             <1>     call    print_msg
                                <1>
                                <1> loc_enter_day_1:
000079E2 30E4                   <1>     xor     ah, ah
000079E4 E83395FFFF             <1>     call    int16h
                                <1>     ; AL = ASCII Code of the Character
000079E9 3C0D                   <1>     cmp     al, 13
                                <1>     ;je     loc_set_date_retn
                                <1>     ; 26/07/2022
000079EB 7404                   <1>     je      short set_date_0
000079ED 3C1B                   <1>     cmp     al, 27
                                <1>     ;je     loc_set_date_retn
                                <1>     ; 26/07/2022
000079EF 7511                   <1>     jne     short set_date_1
                                <1>
                                <1> set_date_0:
                                <1> ;loc_set_date_retn:
000079F1 BE[E7380100]           <1>     mov     esi, nextline
                                <1>     ;call   print_msg
                                <1>     ;retn
                                <1>     ; 26/07/2022
000079F6 E9AFF3FFFF             <1>     jmp     print_msg
                                <1>
                                <1>     ; 26/07/2022
                                <1> loc_set_date_stc_0:
                                <1>     ;xor    bh, bh ; video page 0
000079FB E8C5A9FFFF             <1>     call    beeper ; BEEP !
00007A00 EBE0                   <1>     jmp     short loc_enter_day_1
                                <1>
                                <1>     ; 26/07/2022
                                <1> set_date_1:
00007A02 A2[A52F0100]           <1>     mov     [Day], al
00007A07 3C30                   <1>     cmp     al, '0'
00007A09 72F0                   <1>     jb      short loc_set_date_stc_0
00007A0B 3C33                   <1>     cmp     al, '3'
00007A0D 77EC                   <1>     ja      short loc_set_date_stc_0
                                <1>     ; 13/05/2016
                                <1>     ;mov    bx, 7 ; attribute/color (bl)
                                <1>                   ; video page 0 (bh)
```

```
00007A0F B307                <1>     mov     bl, 7
00007A11 E8C5A8FFFF          <1>     call    _write_tty
                             <1> loc_enter_day_2:
00007A16 30E4                <1>     xor     ah, ah
00007A18 E8FF94FFFF          <1>     call    int16h
                             <1>     ; AL = ASCII Code of the Character
00007A1D 3C1B                <1>     cmp     al, 27
                             <1>         ;je loc_set_date_retn
                             <1>     ; 26/07/2022
00007A1F 74D0                <1>     je      short set_date_0
00007A21 A2[A62F0100]        <1>     mov     [Day+1], al
00007A26 3C30                <1>     cmp     al, '0'
00007A28 7211                <1>         jb      short loc_set_date_stc_1
00007A2A 3C39                <1>     cmp     al, '9'
00007A2C 770D                <1>         ja      short loc_set_date_stc_1
00007A2E 803D[A52F0100]33    <1>     cmp     byte [Day], '3'
00007A35 7219                <1>     jb      short pass_set_day_31
00007A37 3C31                <1>     cmp     al, '1'
                             <1>         ;ja loc_set_date_stc_1
                             <1>     ; 26/07/2022
00007A39 7615                <1>     jna     short pass_set_day_31
                             <1>
                             <1>     ; 26/07/2022
                             <1> loc_set_date_stc_1:
00007A3B E8E4010000          <1>     call    check_for_backspace
00007A40 7407                <1>     je      short loc_set_date_bs_1
                             <1>     ;xor    bh, bh ; video page 0
00007A42 E87EA9FFFF          <1>     call    beeper ; BEEP !
00007A47 EBCD                <1>     jmp     short loc_enter_day_2
                             <1> loc_set_date_bs_1:
00007A49 E8C4010000          <1>     call    write_backspace
00007A4E EB92                <1>     jmp     short loc_enter_day_1
                             <1>
                             <1> pass_set_day_31:
                             <1>     ; 13/05/2016
                             <1>     ;mov    bx, 7 ; attribute/color (bl)
                             <1>                   ; video page 0 (bh)
00007A50 B307                <1>     mov     bl, 7
00007A52 E884A8FFFF          <1>     call    _write_tty
                             <1> loc_enter_separator_1:
00007A57 28E4                <1>     sub     ah, ah ; 0
00007A59 E8BE94FFFF          <1>     call    int16h
                             <1>     ; AL = ASCII Code of the Character
00007A5E 3C1B                <1>     cmp     al, 27
                             <1>         ;je loc_set_date_retn
                             <1>     ; 26/07/2022
00007A60 748F                <1>     je      short set_date_0
00007A62 3C2D                <1>     cmp     al, '-'
00007A64 7443                <1>     je      short pass_set_date_separator_1
00007A66 3C2F                <1>     cmp     al, '/'
                             <1>     ;jne    loc_set_date_stc_2
                             <1>     ; 26/07/2022
00007A68 743F                <1>     je      short pass_set_date_separator_1
                             <1>
                             <1>     ; 26/07/2022
                             <1> loc_set_date_stc_2:
00007A6A E8B5010000          <1>     call    check_for_backspace
00007A6F 7407                <1>     je      short loc_set_date_bs_2
                             <1>     ;xor    bh, bh ; video page 0
00007A71 E84FA9FFFF          <1>     call    beeper ; BEEP !
00007A76 EBDF                <1>     jmp     short loc_enter_separator_1
                             <1> loc_set_date_bs_2:
00007A78 E895010000          <1>     call    write_backspace
00007A7D EB97                <1>     jmp     short loc_enter_day_2
                             <1>
                             <1>     ; 26/07/2022
                             <1> loc_set_date_stc_3:
00007A7F E8A0010000          <1>     call    check_for_backspace
00007A84 7407                <1>     je      short loc_set_date_bs_3
                             <1>     ;xor    bh, bh ; video page 0
00007A86 E83AA9FFFF          <1>     call    beeper ; BEEP !
00007A8B EB23                <1>     jmp     short loc_enter_month_1
                             <1> loc_set_date_bs_3:
00007A8D E880010000          <1>     call    write_backspace
00007A92 EBC3                <1>     jmp     short loc_enter_separator_1
                             <1>
                             <1>     ; 26/07/2022
                             <1> loc_set_date_stc_4:
00007A94 E88B010000          <1>     call    check_for_backspace
00007A99 7407                <1>     je      short loc_set_date_bs_4
                             <1>     ;xor    bh, bh ; video page 0
00007A9B E825A9FFFF          <1>     call    beeper ; BEEP !
00007AA0 EB2D                <1>     jmp     short loc_enter_month_2
                             <1> loc_set_date_bs_4:
00007AA2 E86B010000          <1>     call    write_backspace
00007AA7 EB07                <1>     jmp     short loc_enter_month_1
                             <1>
                             <1> pass_set_date_separator_1:
                             <1>     ; 13/05/2016
                             <1>     ;mov    bx, 7 ; attribute/color (bl)
                             <1>                   ; video page 0 (bh)
00007AA9 B307                <1>     mov     bl, 7
00007AAB E82BA8FFFF          <1>     call    _write_tty
                             <1> loc_enter_month_1:
00007AB0 30E4                <1>     xor     ah, ah ; 0
00007AB2 E86594FFFF          <1>     call    int16h
                             <1>     ; AL = ASCII Code of the Character
00007AB7 3C1B                <1>     cmp     al, 27
                             <1>         ;je loc_set_date_retn
                             <1>     ; 26/07/2022
                             <1>     ;je     short loc_set_date_ok
                             <1>     ; 07/08/2022
00007AB9 741F                <1>     je      short jmp_loc_set_date_ok
00007ABB A2[A82F0100]        <1>     mov     [Month], al
00007AC0 3C30                <1>     cmp     al, '0'
00007AC2 72BB                <1>         jb      short loc_set_date_stc_3
00007AC4 3C31                <1>     cmp     al, '1'
00007AC6 77B7                <1>         ja  short loc_set_date_stc_3
                             <1>     ; 13/05/2016
                             <1>     ;mov    bx, 7 ; attribute/color (bl)
                             <1>                   ; video page 0 (bh)
00007AC8 B307                <1>     mov     bl, 7
00007ACA E80CA8FFFF          <1>     call    _write_tty
                             <1> loc_enter_month_2:
00007ACF 30E4                <1>     xor     ah, ah
00007AD1 E84694FFFF          <1>     call    int16h
                             <1>     ; AL = ASCII Code of the Character
00007AD6 3C1B                <1>     cmp     al, 27
                             <1>         ;;je        loc_set_date_retn
                             <1>     ; 26/07/2022
                             <1>     ;je     short loc_set_date_ok
                             <1>     ; 07/08/2022
00007AD8 7505                <1>     jne     short loc_enter_month_3
                             <1> jmp_loc_set_date_ok:
```

```
00007ADA E996000000          <1>     jmp     loc_set_date_ok
                             <1> loc_enter_month_3:
00007ADF A2[A92F0100]        <1>     mov     [Month+1], al
00007AE4 3C30                <1>     cmp     al, '0'
00007AE6 72AC                <1>         jb  short loc_set_date_stc_4
00007AE8 3C39                <1>     cmp     al, '9'
00007AEA 77A8                <1>         ja  short loc_set_date_stc_4
00007AEC 803D[A82F0100]31    <1>     cmp     byte [Month], '1'
00007AF3 7204                <1>     jb      short pass_set_month_12
00007AF5 3C32                <1>     cmp     al, '2'
00007AF7 779B                <1>         ja  short 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)
00007AF9 B307                <1>     mov     bl, 7
00007AFB E8DBA7FFFF          <1>     call    _write_tty
                             <1> loc_enter_separator_2:
00007B00 28E4                <1>     sub     ah, ah
00007B02 E81594FFFF          <1>     call    int16h
                             <1>     ; AL = ASCII Code of the Character
00007B07 3C1B                <1>     cmp     al, 27
                             <1>         ;je loc_set_date_retn
                             <1>     ; 26/07/2022
00007B09 746A                <1>     je      short loc_set_date_ok
00007B0B 3C2D                <1>     cmp     al, '-'
00007B0D 7404                <1>     je      short pass_set_date_separator_2
00007B0F 3C2F                <1>     cmp     al, '/'
00007B11 756C                <1>         jne short 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)
00007B13 B307                <1>     mov     bl, 7
00007B15 E8C1A7FFFF          <1>     call    _write_tty
                             <1> loc_enter_year_1:
00007B1A 30E4                <1>     xor     ah, ah
00007B1C E8FB93FFFF          <1>     call    int16h
                             <1>     ; AL = ASCII Code of the Character
00007B21 3C1B                <1>     cmp     al, 27
                             <1>         ;je loc_set_date_retn
                             <1>     ; 26/07/2022
00007B23 7450                <1>     je      short loc_set_date_ok
00007B25 A2[AD2F0100]        <1>     mov     [Year], al
00007B2A 3C30                <1>     cmp     al, '0'
00007B2C 726C                <1>         jb  short loc_set_date_stc_6
00007B2E 3C39                <1>     cmp     al, '9'
00007B30 7768                <1>         ja  short loc_set_date_stc_6
                             <1>     ; 13/05/2016
                             <1>     ;mov    bx, 7 ; attribute/color (bl)
                             <1>                   ; video page 0 (bh)
00007B32 B307                <1>     mov     bl, 7
00007B34 E8A2A7FFFF          <1>     call    _write_tty
                             <1> loc_enter_year_2:
00007B39 30E4                <1>     xor     ah, ah
00007B3B E8DC93FFFF          <1>     call    int16h
                             <1>     ; AL = ASCII Code of the Character
00007B40 3C1B                <1>     cmp     al, 27
                             <1>     ;je     short loc_set_date_retn
                             <1>     ; 26/07/2022
00007B42 7431                <1>     je      short loc_set_date_ok
00007B44 A2[AE2F0100]        <1>     mov     byte [Year+1], al
00007B49 3C30                <1>     cmp     al, '0'
00007B4B 7268                <1>         jb  short loc_set_date_stc_7
00007B4D 3C39                <1>     cmp     al, '9'
00007B4F 7764                <1>         ja  short loc_set_date_stc_7
                             <1>     ; 13/05/2016
                             <1>     ;mov    bx, 7 ; attribute/color (bl)
                             <1>                   ; video page 0 (bh)
00007B51 B307                <1>     mov     bl, 7
00007B53 E883A7FFFF          <1>     call    _write_tty
                             <1> loc_set_date_get_lchar_again:
00007B58 28E4                <1>     sub     ah, ah ; 0
00007B5A E8BD93FFFF          <1>     call    int16h
                             <1>     ; AL = ASCII Code of the Character
00007B5F 3C0D                <1>     cmp     al, 13 ; ENTER key
00007B61 746D                <1>     je      short loc_set_date_progress
00007B63 3C1B                <1>     cmp     al, 27 ; ESC key
                             <1>     ;je     short loc_set_date_retn
                             <1>     ; 26/07/2022
00007B65 740E                <1>     je      short loc_set_date_ok
                             <1>     ;
00007B67 E8B8000000          <1>     call    check_for_backspace
00007B6C 75EA                <1>     jne     short loc_set_date_get_lchar_again
                             <1>
                             <1> loc_set_date_bs_8:
00007B6E E89F000000          <1>     call    write_backspace
00007B73 EBC4                <1>     jmp     short loc_enter_year_2
                             <1>
                             <1> loc_set_date_ok:
                             <1> ;loc_set_date_retn:
00007B75 BE[E7380100]        <1>     mov     esi, nextline
                             <1>     ;call   print_msg
                             <1>     ;retn
                             <1>     ; 26/07/2022
00007B7A E92BF2FFFF          <1>     jmp     print_msg
                             <1>
                             <1>     ; 26/07/2022
                             <1> ;loc_set_date_stc_0:
                             <1> ;   ;xor    bh, bh ; video page 0
                             <1> ;   call    beeper ; BEEP !
                             <1> ;   jmp     loc_enter_day_1
                             <1> ;loc_set_date_stc_1:
                             <1> ;   call    check_for_backspace
                             <1> ;   je      short loc_set_date_bs_1
                             <1> ;   ;xor    bh, bh ; video page 0
                             <1> ;   call    beeper ; BEEP !
                             <1> ;   jmp     loc_enter_day_2
                             <1> ;loc_set_date_bs_1:
                             <1> ;   call    write_backspace
                             <1> ;   jmp     loc_enter_day_1
                             <1> ;loc_set_date_stc_2:
                             <1> ;   call    check_for_backspace
                             <1> ;   je      short loc_set_date_bs_2
                             <1> ;   ;xor    bh, bh ; video page 0
                             <1> ;   call    beeper ; BEEP !
                             <1> ;   jmp     loc_enter_separator_1
                             <1> ;loc_set_date_bs_2:
                             <1> ;   call    write_backspace
                             <1> ;   jmp     loc_enter_day_2
                             <1> ;loc_set_date_stc_3:
                             <1> ;   call    check_for_backspace
                             <1> ;   je      short loc_set_date_bs_3
                             <1> ;   ;xor    bh, bh ; video page 0
```

```
                        <1> ;   call    beeper ; BEEP !
                        <1> ;   jmp     loc_enter_month_1
                        <1> ;loc_set_date_bs_3:
                        <1> ;   call    write_backspace
                        <1> ;   jmp     loc_enter_separator_1
                        <1> ;loc_set_date_stc_4:
                        <1> ;   call    check_for_backspace
                        <1> ;   je      short loc_set_date_bs_4
                        <1> ;   ;xor    bh, bh ; video page 0
                        <1> ;   call    beeper ; BEEP !
                        <1> ;   jmp     loc_enter_month_2
                        <1> ;loc_set_date_bs_4:
                        <1> ;   call    write_backspace
                        <1> ;   jmp     loc_enter_month_1
                        <1>
                        <1>     ; 26/07/2022
                        <1> loc_set_date_stc_5:
00007B7F E8A0000000     <1>     call    check_for_backspace
00007B84 740A           <1>     je      short loc_set_date_bs_5
                        <1>     ;xor    bh, bh ; video page 0
00007B86 E83AA8FFFF     <1>     call    beeper ; BEEP !
00007B8B E970FFFFFF     <1>     jmp     loc_enter_separator_2
                        <1> loc_set_date_bs_5:
00007B90 E87D000000     <1>     call    write_backspace
00007B95 E935FFFFFF     <1>     jmp     loc_enter_month_2
                        <1> loc_set_date_stc_6:
00007B9A E885000000     <1>     call    check_for_backspace
00007B9F 740A           <1>         je      short loc_set_date_bs_6
                        <1>     ;xor    bh, bh ; video page 0
00007BA1 E81FA8FFFF     <1>     call    beeper ; BEEP !
00007BA6 E96FFFFFFF     <1>     jmp     loc_enter_year_1
                        <1> loc_set_date_bs_6:
00007BAB E862000000     <1>     call    write_backspace
00007BB0 E94BFFFFFF     <1>     jmp     loc_enter_separator_2
                        <1> loc_set_date_stc_7:
00007BB5 E86A000000     <1>     call    check_for_backspace
00007BBA 740A           <1>     je      short loc_set_date_bs_7
                        <1>     ;xor    bh, bh ; video page 0
00007BBC E804A8FFFF     <1>     call    beeper ; BEEP !
00007BC1 E973FFFFFF     <1>     jmp     loc_enter_year_2
                        <1> loc_set_date_bs_7:
00007BC6 E847000000     <1>     call    write_backspace
00007BCB E94AFFFFFF     <1>     jmp     loc_enter_year_1
                        <1>
                        <1> loc_set_date_progress:
                        <1>     ; Get Current Date
                        <1>     ;mov    ah, 04h
                        <1>     ;call   int1Ah
00007BD0 E8E7E7FFFF     <1>     call    RTC_40  ; GET RTC DATE
                        <1>     ; CH = century (in BCD)
                        <1>
00007BD5 66A1[AD2F0100] <1>     mov     ax, [Year]
00007BDB 662D3030       <1>     sub     ax, '00'
00007BDF C0E004         <1>     shl     al, 4 ; * 16
00007BE2 88C1           <1>     mov     cl, al
00007BE4 00E1           <1>     add     cl, ah
00007BE6 66A1[A82F0100] <1>     mov     ax, [Month]
00007BEC 662D3030       <1>     sub     ax, '00'
00007BF0 C0E004         <1>     shl     al, 4 ; * 16
00007BF3 88C6           <1>     mov     dh, al
00007BF5 00E6           <1>     add     dh, ah
00007BF7 66A1[A52F0100] <1>     mov     ax, [Day]
00007BFD 662D3030       <1>     sub     ax, '00'
00007C01 C0E004         <1>     shl     al, 4 ; * 16
00007C04 88C2           <1>     mov     dl, al
00007C06 00E2           <1>     add     dl, ah
                        <1>
                        <1>     ;mov    ah, 05h
                        <1>     ;call   int1Ah
00007C08 E8D5E7FFFF     <1>     call    RTC_50  ; SET RTC DATE
                        <1>
                        <1> ;loc_set_date_retn:
                        <1> ;   mov     esi, nextline
                        <1> ;   ;call   print_msg
                        <1> ;   ;retn
                        <1> ;   ; 26/07/2022
                        <1> ;   jmp     print_msg
                        <1>
                        <1>     ; 26/07/2022
00007C0D E963FFFFFF     <1>     jmp     loc_set_date_ok
                        <1>
                        <1> write_backspace:
                        <1>     ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
00007C12 B008           <1>     mov     al, 08h ; BACKSPACE
                        <1>     ; 13/05/2016
00007C14 66BB0700       <1>     mov     bx, 7 ; bl = attribute/color
                        <1>                   ; bh = video page = 0
00007C18 E8BEA6FFFF     <1>     call    _write_tty
00007C1D B020           <1>     mov     al, 20h ; BLANK/SPACE char
                        <1>     ;mov    bx, 7 ; attribute/color
                        <1>     ;call   _write_c_current
                        <1>     ;retn
00007C1F E936A6FFFF     <1>     jmp     _write_c_current
                        <1>
                        <1> check_for_backspace:
                        <1>     ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
00007C24 663D080E       <1>     cmp     ax, 0E08h
00007C28 7410           <1>     je      short cfbs_retn
00007C2A 663DE04B       <1>     cmp     ax, 4BE0h
00007C2E 740A           <1>     je      short cfbs_retn
00007C30 663D004B       <1>     cmp     ax, 4B00h
00007C34 7404           <1>     je      short cfbs_retn
00007C36 663DE053       <1>     cmp     ax, 53E0h
                        <1> cfbs_retn:
00007C3A C3             <1>     retn
                        <1>
                        <1> show_time:
                        <1>     ; 26/07/2022 (TRDOS 386 Kernel v2.0.5)
                        <1>     ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>         ; 2004-2005
                        <1>
                        <1>     ;mov    ah, 02h
                        <1>     ;call   int1Ah
00007C3B E811E7FFFF     <1>     call    RTC_20  ; GET RTC TIME
                        <1>
00007C40 88E8           <1>     mov     al, ch
00007C42 E8C892FFFF     <1>     call    bcd_to_ascii
00007C47 66A3[D32F0100] <1>     mov     [Hour], ax
                        <1>
00007C4D 88C8           <1>     mov     al, cl
00007C4F E8BB92FFFF     <1>     call    bcd_to_ascii
00007C54 66A3[D62F0100] <1>     mov     [Minute], ax
                        <1>
00007C5A 88F0           <1>     mov     al, dh
```

```
 977 00007C5C E8AE92FFFF          <1>     call    bcd_to_ascii
 978 00007C61 66A3[D92F0100]      <1>     mov     [Second], ax
 979                              <1>
 980 00007C67 BE[C32F0100]        <1>     mov     esi, Msg_Show_Time
 981                              <1>     ;call   print_msg
 982                              <1>     ;retn
 983                              <1>     ; 26/07/2022
 984 00007C6C E939F1FFFF          <1>     jmp     print_msg
 985                              <1>
 986                              <1> set_time:
 987                              <1>     ; 26/07/2022 (TRDOS 386 Kernel v2.0.5)
 988                              <1>     ; 13/05/2016
 989                              <1>     ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
 990                              <1>         ; 2004-2005
 991                              <1>
 992 00007C71 BE[B22F0100]        <1>     mov     esi, Msg_Enter_Time
 993 00007C76 E82FF1FFFF          <1>     call    print_msg
 994                              <1>
 995                              <1> loc_enter_hour_1:
 996 00007C7B 30E4                <1>     xor     ah, ah
 997 00007C7D E89A92FFFF          <1>     call    int16h
 998                              <1>     ; AL = ASCII Code of the Character
 999 00007C82 3C0D                <1>     cmp     al, 13 ; ENTER key
1000 00007C84 742D                <1>         je  short loc_set_time_retn
1001 00007C86 3C1B                <1>     cmp     al, 27 ; ESC key
1002 00007C88 7429                <1>         je  short loc_set_time_retn
1003                              <1> set_time_0:
1004 00007C8A A2[D32F0100]        <1>     mov     [Hour], al
1005 00007C8F 3C30                <1>     cmp     al, '0'
1006 00007C91 7204                <1>         jb  short loc_set_time_stc_0
1007 00007C93 3C32                <1>     cmp     al, '2'
1008                              <1>     ;ja     loc_set_time_stc_0
1009                              <1>     ; 26/07/2022
1010 00007C95 7626                <1>     jna     short set_time_1
1011                              <1>
1012                              <1>     ; 26/07/2022
1013                              <1> loc_set_time_stc_0:
1014                              <1>     ;xor    bh, bh ; video page 0
1015 00007C97 E829A7FFFF          <1>     call    beeper ; BEEP !
1016 00007C9C EBDD                <1>     jmp     short loc_enter_hour_1
1017                              <1>
1018                              <1> loc_set_time_stc_1:
1019 00007C9E E881FFFFFF          <1>     call    check_for_backspace
1020 00007CA3 7407                <1>     je      short loc_set_time_bs_1
1021                              <1>     ;xor    bh, bh ; video page 0
1022 00007CA5 E81BA7FFFF          <1>     call    beeper ; BEEP !
1023 00007CAA EB18                <1>     jmp     short loc_enter_hour_2
1024                              <1> loc_set_time_bs_1:
1025 00007CAC E861FFFFFF          <1>     call    write_backspace
1026 00007CB1 EBC8                <1>     jmp     short loc_enter_hour_1
1027                              <1>
1028                              <1>     ; 26/07/2022
1029                              <1> loc_set_time_retn:
1030 00007CB3 BE[E7380100]        <1>     mov     esi, nextline
1031                              <1>     ;call   print_msg
1032                              <1>     ;retn
1033 00007CB8 E9EDF0FFFF          <1>     jmp     print_msg
1034                              <1>
1035                              <1> set_time_1:
1036                              <1>     ; 13/05/2016
1037                              <1>     ;mov    bx, 7 ; attribute/color (bl)
1038                              <1>                   ; video page 0 (bh)
1039 00007CBD B307                <1>     mov     bl, 7
1040 00007CBF E817A6FFFF          <1>     call    _write_tty
1041                              <1> loc_enter_hour_2:
1042 00007CC4 30E4                <1>     xor     ah, ah
1043 00007CC6 E85192FFFF          <1>     call    int16h
1044                              <1>     ; AL = ASCII Code of the Character
1045 00007CCB 3C1B                <1>     cmp     al, 27
1046 00007CCD 74E4                <1>     je      short loc_set_time_retn
1047 00007CCF A2[D42F0100]        <1>     mov     [Hour+1], al
1048 00007CD4 3C30                <1>     cmp     al, '0'
1049 00007CD6 72C6                <1>     jb      short loc_set_time_stc_1
1050 00007CD8 3C39                <1>     cmp     al, '9'
1051 00007CDA 77C2                <1>     ja      short loc_set_time_stc_1
1052 00007CDC 803D[D32F0100]32    <1>         cmp     byte [Hour], '2'
1053 00007CE3 7204                <1>     jb      short pass_set_time_24
1054 00007CE5 3C34                <1>     cmp     al, '4'
1055 00007CE7 77B5                <1>     ja      short loc_set_time_stc_1
1056                              <1> pass_set_time_24:
1057                              <1>     ; 13/05/2016
1058                              <1>     ;mov    bx, 7 ; attribute/color (bl)
1059                              <1>                   ; video page 0 (bh)
1060 00007CE9 B307                <1>     mov     bl, 7
1061 00007CEB E8EBA5FFFF          <1>     call    _write_tty
1062                              <1> loc_enter_time_separator_1:
1063 00007CF0 28E4                <1>     sub     ah, ah ; 0
1064 00007CF2 E82592FFFF          <1>     call    int16h
1065                              <1>     ; AL = ASCII Code of the Character
1066 00007CF7 3C1B                <1>     cmp     al, 27
1067 00007CF9 74B8                <1>     je      short loc_set_time_retn
1068 00007CFB 3C3A                <1>     cmp     al, ':'
1069                              <1>     ;jne    loc_set_time_stc_2
1070                              <1>     ; 26/07/2022
1071 00007CFD 7415                <1>     je      short set_time_2
1072                              <1>
1073                              <1>     ; 26/07/2022
1074                              <1> loc_set_time_stc_2:
1075 00007CFF E820FFFFFF          <1>     call    check_for_backspace
1076 00007D04 7407                <1>     je      short loc_set_time_bs_2
1077                              <1>     ;xor    bh, bh ; video page 0
1078 00007D06 E8BAA6FFFF          <1>     call    beeper ; BEEP !
1079 00007D0B EBE3                <1>     jmp     short loc_enter_time_separator_1
1080                              <1> loc_set_time_bs_2:
1081 00007D0D E800FFFFFF          <1>     call    write_backspace
1082 00007D12 EBB0                <1>     jmp     short loc_enter_hour_2
1083                              <1>
1084                              <1> set_time_2:
1085                              <1>     ; 13/05/2016
1086                              <1>     ;mov    bx, 7 ; attribute/color (bl)
1087                              <1>                   ; video page 0 (bh)
1088 00007D14 B307                <1>     mov     bl, 7
1089 00007D16 E8C0A5FFFF          <1>     call    _write_tty
1090                              <1> loc_enter_minute_1:
1091 00007D1B 30E4                <1>     xor     ah, ah
1092 00007D1D E8FA91FFFF          <1>     call    int16h
1093                              <1>     ; AL = ASCII Code of the Character
1094 00007D22 3C1B                <1>     cmp     al, 27
1095 00007D24 748D                <1>     je      short loc_set_time_retn
1096 00007D26 A2[D62F0100]        <1>     mov     [Minute], al
1097 00007D2B 3C30                <1>     cmp     al, '0'
1098 00007D2D 7204                <1>     jb      short loc_set_time_stc_3
1099 00007D2F 3C35                <1>     cmp     al, '5'
1100                              <1>     ;ja     loc_set_time_stc_3
```

```
                                <1>     ; 26/07/2022
00007D31 7615                   <1>     jna     short set_time_3
                                <1>
                                <1>     ; 26/07/2022
                                <1> loc_set_time_stc_3:
00007D33 E8ECFEFFFF             <1>     call    check_for_backspace
00007D38 7407                   <1>     je      short loc_set_time_bs_3
                                <1>     ;xor    bh, bh ; video page 0
00007D3A E886A6FFFF             <1>     call    beeper ; BEEP !6
00007D3F EBDA                   <1>     jmp     short loc_enter_minute_1
                                <1> loc_set_time_bs_3:
00007D41 E8CCFEFFFF             <1>     call    write_backspace
00007D46 EBA8                   <1>     jmp     short loc_enter_time_separator_1
                                <1>
                                <1> set_time_3:
                                <1>     ; 13/05/2016
                                <1>     ;mov    bx, 7 ; attribute/color (bl)
                                <1>                   ; video page 0 (bh)
00007D48 B307                   <1>     mov     bl, 7
00007D4A E88CA5FFFF             <1>     call    _write_tty
                                <1> loc_enter_minute_2:
00007D4F 30E4                   <1>     xor     ah, ah
00007D51 E8C691FFFF             <1>     call    int16h
                                <1>     ; AL = ASCII Code of the Character
00007D56 3C1B                   <1>     cmp     al, 27
                                <1>     ;je     short loc_set_time_retn
                                <1>     ; 07/08/2022
00007D58 7505                   <1>     jne     short loc_enter_minute_3
00007D5A E954FFFFFF             <1>     jmp     loc_set_time_retn
                                <1> loc_enter_minute_3:
00007D5F A2[D72F0100]           <1>     mov     [Minute+1], al
00007D64 3C30                   <1>     cmp     al, '0'
00007D66 7204                   <1>     jb      short loc_set_time_stc_4
00007D68 3C39                   <1>     cmp     al, '9'
                                <1>     ;ja     loc_set_time_stc_4
                                <1>     ; 26/07/2022
00007D6A 7615                   <1>     jna     short set_time_4
                                <1>
                                <1>     ; 26/07/2022
                                <1> loc_set_time_stc_4:
00007D6C E8B3FEFFFF             <1>     call    check_for_backspace
00007D71 7407                   <1>     je      short loc_set_time_bs_4
                                <1>     ;xor    bh, bh ; video page 0
00007D73 E84DA6FFFF             <1>     call    beeper ; BEEP !
00007D78 EBD5                   <1>     jmp     short loc_enter_minute_2
                                <1> loc_set_time_bs_4:
00007D7A E893FEFFFF             <1>     call    write_backspace
00007D7F EB9A                   <1>     jmp     short loc_enter_minute_1
                                <1>
                                <1> set_time_4:
                                <1>     ; 13/05/2016
                                <1>     ;mov    bx, 7 ; attribute/color (bl)
                                <1>                   ; video page 0 (bh)
00007D81 B307                   <1>     mov     bl, 7
00007D83 E853A5FFFF             <1>     call    _write_tty
                                <1> loc_enter_time_separator_2:
00007D88 66C705[D92F0100]30-    <1>     mov     word [Second], 3030h
00007D90 30                     <1>
00007D91 28E4                   <1>     sub     ah, ah
00007D93 E88491FFFF             <1>     call    int16h
                                <1>     ; AL = ASCII Code of the Character
00007D98 3C0D                   <1>     cmp     al, 13
                                <1>         ;je short loc_set_time_progress
                                <1>     ; 07/08/2022
00007D9A 7505                   <1>     jne     short loc_enter_time_separator_3
                                <1> jmp_loc_set_time_progress:
00007D9C E9D4000000             <1>     jmp     loc_set_time_progress
                                <1> loc_enter_time_separator_3:
00007DA1 3C1B                   <1>     cmp     al, 27
                                <1>     ;;je    loc_set_time_retn
                                <1>     ; 26/07/2022
                                <1>     ;je     short loc_set_time_ok
                                <1>     ; 07/08/2022
00007DA3 7505                   <1>     jne     short loc_enter_time_separator_4
00007DA5 E982000000             <1>     jmp     loc_set_time_ok
                                <1> loc_enter_time_separator_4:
00007DAA 3C3A                   <1>     cmp     al, ':'
00007DAC 7563                   <1>         jne short loc_set_time_stc_5
                                <1>
                                <1>     ; 13/05/2016
                                <1>     ;mov    bx, 7 ; attribute/color (bl)
                                <1>                   ; video page 0 (bh)
00007DAE B307                   <1>     mov     bl, 7
00007DB0 E826A5FFFF             <1>     call    _write_tty
                                <1> loc_enter_second_1:
00007DB5 30E4                   <1>     xor     ah, ah
00007DB7 E86091FFFF             <1>     call    int16h
                                <1>     ; AL = ASCII Code of the Character
00007DBC 3C0D                   <1>     cmp     al, 13
                                <1>     ;je     short loc_set_time_progress
                                <1>     ; 07/08/2022
00007DBE 74DC                   <1>     je      short jmp_loc_set_time_progress
00007DC0 3C1B                   <1>     cmp     al, 27
                                <1>     ;;je    loc_set_time_retn
                                <1>     ; 26/07/2022
                                <1>     ;je     short loc_set_time_ok
                                <1>     ; 07/08/2022
00007DC2 7502                   <1>     jne     short loc_enter_second_2
00007DC4 EB66                   <1>     jmp     loc_set_time_ok
                                <1> loc_enter_second_2:
00007DC6 A2[D92F0100]           <1>     mov     [Second], al
00007DCB 3C30                   <1>     cmp     al, '0'
00007DCD 7267                   <1>     jb      short loc_set_time_stc_6
00007DCF 3C35                   <1>     cmp     al, '5'
00007DD1 7763                   <1>     ja      short loc_set_time_stc_6
                                <1>     ; 13/05/2016
                                <1>     ;mov    bx, 7 ; attribute/color (bl)
                                <1>                   ; video page 0 (bh)
00007DD3 B307                   <1>     mov     bl, 7
00007DD5 E801A5FFFF             <1>     call    _write_tty
                                <1> loc_enter_second_3:
00007DDA 30E4                   <1>     xor     ah, ah
00007DDC E83B91FFFF             <1>     call    int16h
                                <1>     ; AL = ASCII Code of the Character
00007DE1 3C1B                   <1>     cmp     al, 27
                                <1>     ;je     short loc_set_time_retn
                                <1>     ; 26/07/2022
00007DE3 7447                   <1>     je      short loc_set_time_ok
00007DE5 3C30                   <1>     cmp     al, '0'
00007DE7 7271                   <1>         jb  short loc_set_time_stc_7
00007DE9 3C39                   <1>     cmp     al, '9'
00007DEB 776D                   <1>         ja  short loc_set_time_stc_7
                                <1>     ; 13/05/2016
                                <1>     ;mov    bx, 7 ; attribute/color (bl)
```

```
                                <1>                     ; video page 0 (bh)
00007DED B307                   <1>     mov     bl, 7
00007DEF E8E7A4FFFF             <1>     call    _write_tty
                                <1> loc_set_time_get_lchar_again:
00007DF4 28E4                   <1>     sub     ah, ah ; 0
00007DF6 E82191FFFF             <1>     call    int16h
                                <1>     ; AL = ASCII Code of the Character
00007DFB 3C0D                   <1>     cmp     al, 13
00007DFD 7476                   <1>     je      short loc_set_time_progress
00007DFF 3C1B                   <1>     cmp     al, 27
                                <1>     ;je     short loc_set_time_retn
                                <1>     ; 07/08/2022
00007E01 7429                   <1>     je      short loc_set_time_ok
                                <1>     ;
00007E03 E81CFEFFFF             <1>     call    check_for_backspace
00007E08 75EA                   <1>     jne     short loc_set_time_get_lchar_again
                                <1>
                                <1> loc_set_time_bs_8:
00007E0A E803FEFFFF             <1>     call    write_backspace
00007E0F EBC9                   <1>     jmp     short loc_enter_second_3
                                <1>
                                <1> ;   ; 26/07/2022
                                <1> ;loc_set_time_retn:
                                <1> ;   mov     esi, nextline
                                <1> ;   ;call   print_msg
                                <1> ;   ;retn
                                <1> ;   jmp     print_msg
                                <1>
                                <1>     ; 26/07/2022
                                <1> ;loc_set_time_stc_0:
                                <1> ;   ;xor    bh, bh ; video page 0
                                <1> ;   call    beeper ; BEEP !
                                <1> ;   jmp     loc_enter_hour_1
                                <1> ;loc_set_time_stc_1:
                                <1> ;   call    check_for_backspace
                                <1> ;   je      short loc_set_time_bs_1
                                <1> ;   ;xor    bh, bh ; video page 0
                                <1> ;   call    beeper ; BEEP !
                                <1> ;   jmp     loc_enter_hour_2
                                <1> ;loc_set_time_bs_1:
                                <1> ;   call    write_backspace
                                <1> ;   jmp     loc_enter_hour_1
                                <1> ;loc_set_time_stc_2:
                                <1> ;   call    check_for_backspace
                                <1> ;   je      short loc_set_time_bs_2
                                <1> ;   ;xor    bh, bh ; video page 0
                                <1> ;   call    beeper ; BEEP !
                                <1> ;   jmp     loc_enter_time_separator_1
                                <1> ;loc_set_time_bs_2:
                                <1> ;   call    write_backspace
                                <1> ;   jmp     loc_enter_hour_2
                                <1> ;loc_set_time_stc_3:
                                <1> ;   call    check_for_backspace
                                <1> ;   je      short loc_set_time_bs_3
                                <1> ;   ;xor    bh, bh ; video page 0
                                <1> ;   call    beeper ; BEEP !6
                                <1> ;   jmp     loc_enter_minute_1
                                <1> ;loc_set_time_bs_3:
                                <1> ;   call    write_backspace
                                <1> ;   jmp     loc_enter_time_separator_1
                                <1> ;loc_set_time_stc_4:
                                <1> ;   call    check_for_backspace
                                <1> ;   je      short loc_set_time_bs_4
                                <1> ;   ;xor    bh, bh ; video page 0
                                <1> ;   call    beeper ; BEEP !
                                <1> ;   jmp     loc_enter_minute_2
                                <1> ;loc_set_time_bs_4:
                                <1> ;   call    write_backspace
                                <1> ;   jmp     loc_enter_minute_1
                                <1>
                                <1>     ; 26/07/2022
                                <1> loc_set_time_stc_5:
00007E11 E80EFEFFFF             <1>     call    check_for_backspace
00007E16 740A                   <1>     je      short loc_set_time_bs_5
                                <1>     ;xor    bh, bh ; video page 0
00007E18 E8A8A5FFFF             <1>     call    beeper ; BEEP !
00007E1D E966FFFFFF             <1>     jmp     loc_enter_time_separator_2
                                <1> loc_set_time_bs_5:
00007E22 E8EBFDFFFF             <1>     call    write_backspace
00007E27 E923FFFFFF             <1>     jmp     loc_enter_minute_2
                                <1>
                                <1>     ; 26/07/2022
                                <1> loc_set_time_ok:
00007E2C BE[E7380100]           <1>     mov     esi, nextline
                                <1>     ;call   print_msg
                                <1>     ;retn
00007E31 E974EFFFFF             <1>     jmp     print_msg
                                <1>
                                <1>     ; 07/08/2022
                                <1> loc_set_time_stc_6:
00007E36 E8E9FDFFFF             <1>     call    check_for_backspace
00007E3B 7413                   <1>     je      short loc_set_time_bs_6
                                <1>     ;xor    bh, bh ; video page 0
00007E3D E883A5FFFF             <1>     call    beeper ; BEEP !
00007E42 66C705[D92F0100]30-    <1>     mov     word [Second], 3030h
00007E4A 30                     <1>
00007E4B E965FFFFFF             <1>     jmp     loc_enter_second_1
                                <1> loc_set_time_bs_6:
00007E50 E8BDFDFFFF             <1>     call    write_backspace
00007E55 E92EFFFFFF             <1>     jmp     loc_enter_time_separator_2
                                <1> loc_set_time_stc_7:
00007E5A E8C5FDFFFF             <1>     call    check_for_backspace
00007E5F 740A                   <1>     je      short loc_set_time_bs_7
                                <1>     ;xor    bh, bh ; video page 0
00007E61 E85FA5FFFF             <1>     call    beeper ; BEEP !
00007E66 E96FFFFFFF             <1>     jmp     loc_enter_second_3
                                <1> loc_set_time_bs_7:
00007E6B E8A2FDFFFF             <1>     call    write_backspace
00007E70 E940FFFFFF             <1>     jmp     loc_enter_second_1
                                <1>
                                <1> loc_set_time_progress:
                                <1>     ; Get Current Time
                                <1>     ;mov    ah, 02h
                                <1>     ;call   int1Ah
00007E75 E8D7E4FFFF             <1>     call    RTC_20  ; GET RTC TIME
                                <1>     ;DL = Daylight Savings Enable option (0-1)
                                <1>
00007E7A 66A1[D32F0100]         <1>     mov     ax, [Hour]
00007E80 662D3030               <1>     sub     ax, '00'
00007E84 C0E004                 <1>     shl     al, 4 ; * 16
00007E87 88C5                   <1>     mov     ch, al
00007E89 00E5                   <1>     add     ch, ah
00007E8B 66A1[D62F0100]         <1>     mov     ax, [Minute]
00007E91 662D3030               <1>     sub     ax, '00'
```

```
1347 00007E95 C0E004              <1>     shl     al, 4 ; * 16
1348 00007E98 88C1                <1>     mov     cl, al
1349 00007E9A 00E1                <1>     add     cl, ah
1350 00007E9C 66A1[D92F0100]      <1>     mov     ax, [Second]
1351 00007EA2 662D3030            <1>     sub     ax, '00'
1352 00007EA6 C0E004              <1>     shl     al, 4 ; * 16
1353 00007EA9 88C6                <1>     mov     dh, al
1354 00007EAB 00E6                <1>     add     dh, ah
1355                              <1>
1356                              <1>     ;mov    ah, 03h
1357                              <1>     ;call   int1Ah
1358 00007EAD E8CDE4FFFF          <1>     call    RTC_30  ; SET RTC TIME
1359                              <1>
1360                              <1>     ; 26/07/2022
1361 00007EB2 E975FFFFFF          <1>     jmp     loc_set_time_ok
1362                              <1>
1363                              <1> print_volume_info:
1364                              <1>     ; 19/12/2025 - TRDOS 386 v2.0.10
1365                              <1>     ; 01/03/2016
1366                              <1>     ; 08/02/2016
1367                              <1>     ; 06/02/2016
1368                              <1>     ; 04/02/2016
1369                              <1>     ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
1370                              <1>     ; 25/10/2009
1371                              <1>     ;
1372                              <1>     ; "Volume Serial No: "
1373                              <1>     ;
1374                              <1>     ; INPUT  : AL = DOS Drive Number
1375                              <1>     ; OUTPUT : AH = FS Type
1376                              <1>     ;          AL = DOS Drive Name
1377                              <1>     ; CF = 0 -> OK
1378                              <1>     ; CF = 1 -> Drive not ready
1379                              <1>
1380 00007EB7 88C4                <1>     mov     ah, al
1381 00007EB9 28C0                <1>     sub     al, al
1382 00007EBB 0FB7F0              <1>     movzx   esi, ax
1383 00007EBE 81C600010900        <1>     add     esi, Logical_DOSDisks
1384 00007EC4 8A06                <1>     mov     al, [esi]
1385 00007EC6 3C41                <1>     cmp     al, 'A'
1386 00007EC8 7304                <1>     jnb     short loc_pvi_set_vol_name
1387 00007ECA 8A6604              <1>     mov     ah, [esi+LD_FSType]
1388 00007ECD C3                  <1>     retn
1389                              <1>
1390                              <1> loc_pvi_set_vol_name:
1391 00007ECE A2[0D300100]        <1>     mov     [Vol_Drv_Name], al
1392 00007ED3 56                  <1>     push    esi
1393 00007ED4 E833010000          <1>     call    move_volume_name_and_serial_no ;;;
1394 00007ED9 7302                <1>     jnc     short loc_pvi_mvn_ok
1395 00007EDB 5E                  <1>     pop     esi
1396 00007EDC C3                  <1>     retn
1397                              <1>
1398                              <1> loc_pvi_mvn_ok:
1399 00007EDD 8B3424              <1>     mov     esi, [esp]
1400                              <1>
1401                              <1> ; 19/12/2025
1402                              <1> %if 0
1403                              <1>     cmp     byte [esi+LD_FSType], 0A1h
1404                              <1>     jne     short loc_pvi_fat_vol_size
1405                              <1>     mov     eax, [esi+LD_FS_VolumeSize]
1406                              <1>     ; 19/12/2025
1407                              <1>     ;movzx  ebx, word [esi+LD_FS_BytesPerSec]
1408                              <1>     jmp     short loc_vol_size_mul32
1409                              <1> %endif
1410                              <1>
1411                              <1> loc_pvi_fat_vol_size:
1412 00007EE0 8B4670              <1>     mov     eax, [esi+LD_TotalSectors]
1413 00007EE3 8B4670              <1>     mov     eax, [esi+LD_TotalSectors]
1414                              <1>     ; 19/12/2025
1415                              <1>     ;movzx  ebx, word [esi+LD_BPB+BPB_BytsPerSec]
1416                              <1> loc_vol_size_mul32:
1417                              <1>     ; 19/12/2025
1418 00007EE6 BB00020000          <1>     mov     ebx, 512
1419 00007EEB F7E3                <1>     mul     ebx
1420 00007EED 09D2                <1>     or      edx, edx
1421 00007EEF 7507                <1>     jnz     short loc_vol_size_in_kbytes
1422                              <1> loc_vol_size_in_bytes:
1423 00007EF1 B9[EB2F0100]        <1>     mov     ecx, VolSize_Bytes
1424 00007EF6 EB0B                <1>     jmp     short loc_write_vol_size_str
1425                              <1> loc_vol_size_in_kbytes:
1426                              <1>     ;mov    bx, 1024
1427                              <1>     ; 19/12/2025
1428 00007EF8 D1E3                <1>     shl     ebx, 1 ; ebx = 1024
1429 00007EFA F7F3                <1>     div     ebx
1430 00007EFC B9[DE2F0100]        <1>     mov     ecx, VolSize_KiloBytes
1431 00007F01 31D2                <1>     xor     edx, edx ; 0
1432                              <1> loc_write_vol_size_str:
1433 00007F03 890D[787E0100]      <1>     mov     [VolSize_Unit1], ecx
1434                              <1>     ;
1435 00007F09 BF[8E7E0100]        <1>     mov     edi, Vol_Tot_Sec_Str_End
1436                              <1>         ;mov       byte [edi], 0
1437 00007F0E B90A000000          <1>     mov     ecx, 10
1438                              <1> loc_write_vol_size_chr:
1439 00007F13 F7F1                <1>     div     ecx
1440 00007F15 80C230              <1>     add     dl, '0'
1441 00007F18 4F                  <1>     dec     edi
1442 00007F19 8817                <1>     mov     [edi], dl
1443 00007F1B 85C0                <1>     test    eax, eax
1444 00007F1D 7404                <1>     jz      short loc_write_vol_size_str_ok
1445 00007F1F 28D2                <1>     sub     dl, dl ; 0
1446 00007F21 EBF0                <1>     jmp     short loc_write_vol_size_chr
1447                              <1>
1448                              <1> loc_write_vol_size_str_ok:
1449 00007F23 893D[807E0100]      <1>     mov     [Vol_Tot_Sec_Str_Start], edi
1450                              <1>     ;
1451 00007F29 BF[F62F0100]        <1>     mov     edi, Vol_FS_Name
1452 00007F2E 8A4E03              <1>     mov     cl, [esi+LD_FATType]
1453                              <1>
1454                              <1> ; 19/12/2025
1455                              <1> %if 0
1456                              <1>     and     cl, cl ; 0 ?
1457                              <1>     jnz     short loc_write_vol_FAT_str_1
1458                              <1>     mov     word [edi], 'TR'
1459                              <1>     mov     dword [edi+4], ' FS1'
1460                              <1>     ; 19/12/2025
1461                              <1>     ;;movzx ebx, word [esi+LD_FS_BytesPerSec]
1462                              <1>     ;mov    bx, [esi+LD_FS_BytesPerSec]
1463                              <1>     mov     eax, [esi+LD_FS_FreeSectors]
1464                              <1>     jmp     short loc_vol_freespace_mul32
1465                              <1> %endif
1466                              <1>
1467                              <1> loc_write_vol_FAT_str_1:
1468 00007F31 66B83332            <1>     mov     ax, '32' ; FAT32
1469 00007F35 80F902              <1>     cmp     cl, 2 ; [esi+LD_FATType]
1470 00007F38 7708                <1>     ja      short loc_write_vol_FAT_str_2
```

```
00007F3A 66B83132           <1>     mov     ax, '12' ; FAT12
00007F3E 7202               <1>     jb      short loc_write_vol_FAT_str_2
00007F40 B436               <1>     mov     ah, '6'  ; FAT16
                            <1> loc_write_vol_FAT_str_2:
00007F42 C70746415420       <1>     mov     dword [edi], 'FAT '
00007F48 66894704           <1>     mov     word [edi+4], ax
                            <1>     ; 19/12/2025
                            <1>     ;;movzx ebx, word [esi+LD_BPB+BPB_BytsPerSec]
                            <1>     ;mov    bx, [esi+LD_BPB+BPB_BytsPerSec]
00007F4C 8B4674             <1>     mov     eax, [esi+LD_FreeSectors]
                            <1>
                            <1> loc_vol_freespace_recalc0:
                            <1>     ; 01/03/2016
00007F4F 83F8FF             <1>     cmp     eax, 0FFFFFFFFh
00007F52 7209               <1>     jb      short loc_vol_freespace_mul32
                            <1>     ;inc    eax ; 0
                            <1>
                            <1> ; 19/12/2025
                            <1> %if 0
                            <1>     and     cl, cl ; byte [esi+LD_FATType]
                            <1>     jz      short loc_vol_freespace_mul32
                            <1> %endif
                            <1>     ; 19/12/2025
                            <1>     ;push   ebx
00007F54 66BB00FF           <1>     mov     bx, 0FF00h ; recalculate free sectors
00007F58 E804460000         <1>     call    calculate_fat_freespace
                            <1>     ;pop    ebx
                            <1>
                            <1>     ; 19/12/2025
                            <1>     ; (if cf = 1, eax = -1)
                            <1>     ; if eax = -1 , free sector count is invalid
                            <1>
                            <1> loc_vol_freespace_mul32:
00007F5D B9[EB2F0100]       <1>     mov     ecx, VolSize_Bytes
00007F62 BF[9E7E0100]       <1>     mov     edi, Vol_Free_Sectors_Str_End
                            <1>     ;mov    byte [edi], 0
00007F67 40                 <1>     inc     eax ; * ; -1 -> 0
00007F68 7506               <1>     jnz     short loc_vol_freespace_mul32_@
00007F6A 4F                 <1>     dec     edi
00007F6B C6073F             <1>     mov     byte [edi], '?'
00007F6E EB32               <1>     jmp     short loc_write_vol_fspace_str_ok
                            <1>
                            <1> loc_vol_freespace_mul32_@:
00007F70 48                 <1>     dec     eax ; *
                            <1>     ; 19/12/2025
00007F71 BB00020000         <1>     mov     ebx, 512
00007F76 F7E3               <1>     mul     ebx
00007F78 09D2               <1>     or      edx, edx
                            <1>     ;jnz    short loc_vol_fspace_in_kbytes
                            <1> loc_vol_fspace_in_bytes:
                            <1>     ; 19/12/2025
                            <1>     ;mov    ecx, VolSize_Bytes
                            <1>     ;jmp    short loc_write_vol_fspace_str
00007F7A 740B               <1>     jz      short loc_write_vol_fspace_str
                            <1> loc_vol_fspace_in_kbytes:
                            <1>     ;mov    bx, 1024
                            <1>     ; 19/12/2025
00007F7C D1E3               <1>     shl     ebx, 1 ; ebx = 1024
00007F7E F7F3               <1>     div     ebx
00007F80 B9[DE2F0100]       <1>     mov     ecx, VolSize_KiloBytes
00007F85 31D2               <1>     xor     edx, edx ; 0
                            <1> loc_write_vol_fspace_str:
00007F87 890D[7C7E0100]     <1>     mov     [VolSize_Unit2], ecx
                            <1>     ;
                            <1>     ; 19/12/2025
                            <1>     ;mov    edi, Vol_Free_Sectors_Str_End
                            <1>         ;mov       byte [edi], 0
00007F8D B90A000000         <1>     mov     ecx, 10
                            <1> loc_write_vol_fspace_chr:
00007F92 F7F1               <1>     div     ecx
00007F94 80C230             <1>     add     dl, '0'
00007F97 4F                 <1>     dec     edi
00007F98 8817               <1>     mov     [edi], dl
00007F9A 85C0               <1>     test    eax, eax
00007F9C 7404               <1>     jz      short loc_write_vol_fspace_str_ok
00007F9E 28D2               <1>     sub     dl, dl ; 0
00007FA0 EBF0               <1>     jmp     short loc_write_vol_fspace_chr
                            <1>
                            <1> loc_write_vol_fspace_str_ok:
00007FA2 893D[907E0100]     <1>     mov     [Vol_Free_Sectors_Str_Start], edi
                            <1>     ;
00007FA8 BE[F42F0100]       <1>     mov     esi, Volume_in_drive
00007FAD E8F8EDFFFF         <1>     call    print_msg
00007FB2 BE[34300100]       <1>     mov     esi, Vol_Name
00007FB7 E8EEEDFFFF         <1>     call    print_msg
00007FBC BE[E7380100]       <1>     mov     esi, nextline
00007FC1 E8E4EDFFFF         <1>     call    print_msg
                            <1>     ;
00007FC6 BE[95300100]       <1>     mov     esi, Vol_Total_Sector_Header
00007FCB E8DAEDFFFF         <1>     call    print_msg
00007FD0 8B35[807E0100]     <1>     mov     esi, [Vol_Tot_Sec_Str_Start]
00007FD6 E8CFEDFFFF         <1>     call    print_msg
00007FDB 8B35[787E0100]     <1>     mov     esi, [VolSize_Unit1]
00007FE1 E8C4EDFFFF         <1>     call    print_msg
                            <1>     ;
00007FE6 BE[A6300100]       <1>     mov     esi, Vol_Free_Sectors_Header
00007FEB E8BAEDFFFF         <1>     call    print_msg
00007FF0 8B35[907E0100]     <1>     mov     esi, [Vol_Free_Sectors_Str_Start]
00007FF6 E8AFEDFFFF         <1>     call    print_msg
00007FFB 8B35[7C7E0100]     <1>     mov     esi, [VolSize_Unit2]
00008001 E8A4EDFFFF         <1>     call    print_msg
                            <1>     ;
00008006 5E                 <1>     pop     esi
                            <1>
                            <1>     ;mov    ah, [esi+LD_FSType]
                            <1>     ;mov    al, [esi+LD_FATType]
00008007 668B4603           <1>     mov     ax, [esi+LD_FATType]
                            <1>
0000800B C3                 <1>     retn
                            <1>
                            <1> move_volume_name_and_serial_no:
                            <1>     ; 19/12/2025 (TRDOS 386 Kernel v2.0.10)
                            <1>     ; 26/07/2022 (TRDOS 386 Kernel v2.0.5)
                            <1>     ; 08/02/2016  (TRDOS 386 = TRDOS v2.0)
                            <1>     ; this routine will be called by
                            <1>     ; "print_volume_info" and "print_directory"
                            <1>     ; INPUT ->
                            <1>     ;       ESI = Logical DOS drv descripton table address
                            <1>     ; OUTPUT ->
                            <1>     ;       *Volume name will be moved to text area
                            <1>     ;       *Volume serial number will be converted to
                            <1>     ;        text and will be moved to text area
                            <1>     ;   cf = 1 -> invalid/unknown dos drive
                            <1>     ;   cf = 0 -> ecx = 0
```

```
1595                                    <1>     ;
1596                                    <1>     ; (eax, edx, ecx, esi, edi will be changed)
1597                                    <1>
1598                                    <1>     ; 26/07/2022
1599 0000800C 31C9                     <1>     xor     ecx, ecx
1600                                    <1>
1601 0000800E BF[34300100]             <1>     mov     edi, Vol_Name
1602                                    <1>
1603                                    <1>     ;mov    ah, [esi+LD_FSType]
1604                                    <1>     ; 19/12/2025
1605 00008013 8A4603                   <1>     mov     al, [esi+LD_FATType]
1606                                    <1>     ;mov    ax, [esi+LD_FATType]
1607                                    <1>
1608                                    <1> ; 19/12/2025
1609                                    <1> %if 0
1610                                    <1>     cmp     ah, 0A1h
1611                                    <1>     je      short mvn_2
1612                                    <1>     or      ah, ah
1613                                    <1>     jz      short mvn_0
1614                                    <1>     or      al, al
1615                                    <1>     jnz     short mvn_1
1616                                    <1> mvn_0:
1617                                    <1>     mov     al, [esi]
1618                                    <1>     stc
1619                                    <1>     retn
1620                                    <1> %endif
1621                                    <1>
1622                                    <1> mvn_1:
1623 00008016 3C02                     <1>     cmp     al, 2
1624 00008018 7708                     <1>     ja      short mvn_3
1625                                    <1>     ;or     al, al
1626                                    <1>     ;jz     short mvn_2
1627 0000801A 8B462D                   <1>     mov     eax, [esi+LD_BPB+VolumeID]
1628 0000801D 83C631                   <1>     add     esi, LD_BPB+VolumeLabel
1629 00008020 EB06                     <1>     jmp     short mvn_4
1630                                    <1>
1631                                    <1> ; 19/12/2025
1632                                    <1> %if 0
1633                                    <1> mvn_2:
1634                                    <1>     mov     eax, [esi+LD_FS_VolumeSerial]
1635                                    <1>     add     esi, LD_FS_VolumeName
1636                                    <1>     ;mov    ecx, 16
1637                                    <1>     ; 26/07/2022
1638                                    <1>     mov     cl, 16
1639                                    <1>     rep     movsd
1640                                    <1>     jmp     short mvn_5
1641                                    <1> %endif
1642                                    <1>
1643                                    <1> mvn_3:
1644 00008022 8B4649                   <1>     mov     eax, [esi+LD_BPB+FAT32_VolID]
1645 00008025 83C64D                   <1>     add     esi, LD_BPB+FAT32_VolLab
1646                                    <1> mvn_4:
1647                                    <1>     ;mov    ecx, 11
1648                                    <1>     ; 26/07/2022
1649 00008028 B10B                     <1>     mov     cl, 11
1650 0000802A F3A4                     <1>     rep     movsb
1651 0000802C C60700                   <1>     mov     byte [edi], 0
1652                                    <1> mvn_5:
1653                                    <1>     ;mov    [Current_VolSerial], eax
1654 0000802F E8C7C1FFFF               <1>     call    dwordtohex
1655 00008034 8915[89300100]           <1>     mov     [Vol_Serial1], edx
1656 0000803A A3[8E300100]             <1>     mov     [Vol_Serial2], eax
1657                                    <1>     ; ecx = 0
1658 0000803F C3                       <1>     retn
1659                                    <1>
1660                                    <1> get_volume_serial_number:
1661                                    <1>     ; 19/12/2025 (TRDOS 386 Kernel v2.0.10)
1662                                    <1>     ; 19/01/2016 (TRDOS 386 = TRDOS v2.0)
1663                                    <1>     ; 08/08/2010
1664                                    <1>     ;
1665                                    <1>     ; INPUT -> DL = Logical DOS Drive number
1666                                    <1>     ; OUTPUT -> EAX = Volume serial number
1667                                    <1>     ;           BL= FAT Type
1668                                    <1>     ;           BH = Logical DOS drv Number (DL input)
1669                                    <1>     ; cf = 1 -> Drive not ready
1670                                    <1>
1671 00008040 31DB                     <1>     xor     ebx, ebx
1672 00008042 88D7                     <1>     mov     bh, dl
1673 00008044 3815[5C2E0100]           <1>     cmp     [Last_DOS_DiskNo], dl
1674 0000804A 7304                     <1>     jnb     short loc_gvsn_start
1675                                    <1> loc_gvsn_stc_retn:
1676 0000804C 31C0                     <1>     xor     eax, eax
1677 0000804E F9                       <1>     stc
1678 0000804F C3                       <1>         retn
1679                                    <1> loc_gvsn_start:
1680 00008050 56                       <1>     push    esi
1681 00008051 BE00010900               <1>     mov     esi, Logical_DOSDisks
1682 00008056 01DE                     <1>     add     esi, ebx
1683                                    <1> ; 19/12/2025
1684                                    <1> %if 0
1685                                    <1>     mov     bl, [esi+LD_FATType]
1686                                    <1>     and     bl, bl
1687                                    <1>     jz      short loc_gvsn_fs
1688                                    <1> %endif
1689 00008058 80FB02                     <1>     cmp     bl, 2
1690 0000805B 7705                     <1>     ja      short loc_gvsn_fat32
1691                                    <1> loc_gvsn_fat:
1692 0000805D 83C62D                   <1>     add     esi, LD_BPB + VolumeID
1693 00008060 EB03                     <1>     jmp     short loc_gvsn_return
1694                                    <1> loc_gvsn_fat32:
1695 00008062 83C649                   <1>     add     esi, LD_BPB + FAT32_VolID
1696                                    <1> ; 19/12/2025
1697                                    <1> %if 0
1698                                    <1>     jmp     short loc_gvsn_return
1699                                    <1> loc_gvsn_fs:
1700                                    <1>     cmp     byte [esi+LD_FSType], 0A1h
1701                                    <1>     jne     short loc_gvsn_stc_retn
1702                                    <1>     add     esi, LD_FS_VolumeSerial
1703                                    <1> %endif
1704                                    <1>
1705                                    <1> loc_gvsn_return:
1706 00008065 8B06                     <1>     mov     eax, [esi]
1707 00008067 5E                       <1>     pop     esi
1708 00008068 C3                       <1>     retn
1709                                    <1>
1710                                    <1> ; CMD_INTR.ASM [ TRDOS Command Interpreter Procedure ]
1711                                    <1> ; 09/11/2011
1712                                    <1> ; 29/01/2005
1713                                    <1>
1714                                    <1> command_interpreter:
1715                                    <1>     ; 26/07/2022 (TRDOS 386 Kernel v2.0.5)
1716                                    <1>     ; 16/10/2016
1717                                    <1>     ; 12/10/2016
1718                                    <1>     ; 13/05/2016
```

```
1719                                <1>     ; 07/05/2016
1720                                <1>     ; 04/03/2016
1721                                <1>     ; 04/02/2016
1722                                <1>     ; 03/02/2016
1723                                <1>     ; 30/01/2016
1724                                <1>     ; 29/01/2016 (TRDOS 386 = TRDOS 2.0)
1725                                <1>     ; 15/09/2011
1726                                <1>     ; 29/01/2005
1727                                <1>
1728                                <1>     ; Input: ecx = command word length (CL)
1729                                <1>     ;       CommandBuffer = Command string offset
1730                                <1>
1731 00008069 C605[307F0100]00     <1>     mov     byte [Program_Exit], 0
1732                                <1>
1733                                <1>     ;cmp    cl, 4
1734                                <1>     ;ja     c_6
1735                                <1>     ;jb     c_2
1736                                <1>
1737                                <1>     ; 26/07/2022
1738 00008070 80F903               <1>     cmp     cl, 3
1739 00008073 777C                 <1>     ja      short c_4
1740 00008075 7405                 <1>     je      short c_3
1741 00008077 E9D2010000           <1>     jmp     c_2
1742                                <1> c_3:
1743                                <1> cmp_cmd_dir:
1744 0000807C BF[C52E0100]         <1>     mov     edi, Cmd_Dir
1745 00008081 E8D1030000           <1>     call    cmp_cmd
1746                                <1>     ;jnc    print_directory_list
1747                                <1>     ; 26/07/2022
1748 00008086 7205                 <1>     jc      short cmp_cmd_cls
1749 00008088 E97D040000           <1>     jmp     print_directory_list
1750                                <1>
1751                                <1> cmp_cmd_cls:
1752 0000808D B103                 <1>     mov     cl, 3
1753 0000808F BF[012F0100]         <1>     mov     edi, Cmd_Cls
1754 00008094 E8BE030000           <1>     call    cmp_cmd
1755                                <1>         ;jnc       clear_screen
1756                                <1>     ; 26/07/2022
1757 00008099 7205                 <1>     jc      short cmp_cmd_ver
1758 0000809B E920EDFFFF           <1>     jmp     clear_screen
1759                                <1>
1760                                <1> cmp_cmd_ver:
1761 000080A0 B103                 <1>     mov     cl, 3
1762 000080A2 BF[CF2E0100]         <1>     mov     edi, Cmd_Ver
1763 000080A7 E8AB030000           <1>     call    cmp_cmd
1764 000080AC 720A                 <1>     jc      short cmp_cmd_mem
1765                                <1>
1766 000080AE BE[642E0100]         <1>     mov     esi, mainprog_Version
1767                                <1>     ;call   print_msg
1768 000080B3 E9F2ECFFFF           <1>     jmp     print_msg
1769                                <1>     ;retn
1770                                <1>
1771                                <1> cmp_cmd_mem:
1772 000080B8 B103                 <1>     mov     cl, 3
1773 000080BA BF[372F0100]         <1>     mov     edi, Cmd_Mem
1774 000080BF E893030000           <1>     call    cmp_cmd
1775                                <1>     ;jnc    memory_info
1776                                <1>     ; 26/07/2022
1777 000080C4 7205                 <1>     jc      short cmp_cmd_del
1778 000080C6 E95DC0FFFF           <1>     jmp     memory_info
1779                                <1>
1780                                <1> cmp_cmd_del:
1781 000080CB B103                 <1>     mov     cl, 3
1782 000080CD BF[0A2F0100]         <1>     mov     edi, Cmd_Del
1783 000080D2 E880030000           <1>     call    cmp_cmd
1784                                <1>         ;jnc       delete_file
1785                                <1>     ; 26/07/2022
1786 000080D7 7205                 <1>     jc      short cmp_cmd_set
1787 000080D9 E9E2100000           <1>     jmp     delete_file
1788                                <1>
1789                                <1> cmp_cmd_set:
1790 000080DE B103                 <1>     mov     cl, 3
1791 000080E0 BF[FD2E0100]         <1>     mov     edi, Cmd_Set
1792 000080E5 E86D030000           <1>     call    cmp_cmd
1793                                <1>     ;jnc    set_get_env
1794                                <1>     ; 26/07/2022
1795 000080EA 720F                 <1>     jc      short cmp_cmd_run
1796 000080EC E9F1180000           <1>     jmp     set_get_env
1797                                <1>
1798                                <1>     ; 07/08/2022
1799                                <1> c_4:
1800                                <1>     ; 26/07/2022
1801 000080F1 80F904               <1>     cmp     cl, 4
1802 000080F4 741D                 <1>     je      short cmp_cmd_4
1803 000080F6 E937020000           <1>     jmp     c_6
1804                                <1>
1805                                <1> cmp_cmd_run:
1806 000080FB B103                 <1>     mov     cl, 3
1807 000080FD BF[F92E0100]         <1>     mov     edi, Cmd_Run
1808 00008102 E850030000           <1>     call    cmp_cmd
1809                                <1>     ; 07/05/2016
1810                                <1>         ;jc cmp_cmd_external
1811                                <1>     ; 26/07/2022
1812 00008107 7305                 <1>     jnc     short c3_run
1813 00008109 E92C030000           <1>     jmp     cmp_cmd_external
1814                                <1> c3_run:
1815 0000810E E9D71E0000           <1>     jmp     load_and_execute_file
1816                                <1>
1817                                <1> cmp_cmd_4:
1818                                <1>     ; 26/07/2022
1819                                <1> cmp_cmd_exit:
1820 00008113 BF[D32E0100]         <1>     mov     edi, Cmd_Exit
1821 00008118 E83A030000           <1>     call    cmp_cmd
1822 0000811D 7208                 <1>     jc      short cmp_cmd_date
1823                                <1>
1824 0000811F C605[307F0100]01     <1>         mov     byte [Program_Exit], 1
1825 00008126 C3                   <1>         retn
1826                                <1>
1827                                <1> cmp_cmd_date:
1828 00008127 B104                 <1>     mov     cl, 4
1829 00008129 BF[EF2E0100]         <1>     mov     edi, Cmd_Date
1830 0000812E E824030000           <1>     call    cmp_cmd
1831 00008133 720A                 <1>         jc  short cmp_cmd_time
1832                                <1>
1833 00008135 E85BF8FFFF           <1>     call    show_date
1834                                <1>     ;call   set_date
1835                                <1>     ;retn
1836                                <1>     ; 26/07/2022
1837 0000813A E999F8FFFF           <1>     jmp     set_date
1838                                <1>
1839                                <1> cmp_cmd_time:
1840 0000813F B104                 <1>     mov     cl, 4
1841 00008141 BF[F42E0100]         <1>     mov     edi, Cmd_Time
1842 00008146 E80C030000           <1>     call    cmp_cmd
```

```
1843 0000814B 720A                <1>     jc      short cmp_cmd_show
1844                              <1>
1845 0000814D E8E9FAFFFF          <1>     call    show_time
1846                              <1>     ;call   set_time
1847                              <1>     ;retn
1848                              <1>     ; 26/07/2022
1849 00008152 E91AFBFFFF          <1>     jmp     set_time
1850                              <1>
1851                              <1> cmp_cmd_show:
1852 00008157 B104                <1>     mov     cl, 4
1853 00008159 BF[052F0100]        <1>     mov     edi, Cmd_Show
1854 0000815E E8F4020000          <1>     call    cmp_cmd
1855                              <1>         ;jnc       show_file
1856                              <1>     ; 26/07/2022
1857 00008163 7205                <1>     jc      short cmp_cmd_echo
1858 00008165 E971090000          <1>     jmp     show_file
1859                              <1>
1860                              <1> cmp_cmd_echo:
1861 0000816A B104                <1>     mov     cl, 4
1862 0000816C BF[412F0100]        <1>     mov     edi, Cmd_Echo
1863 00008171 E8E1020000          <1>     call    cmp_cmd
1864 00008176 7224                <1>     jc      short cmp_cmd_copy
1865                              <1>
1866                              <1>     ; 22/11/2017
1867                              <1>     ; AL = 0
1868 00008178 803E20              <1>     cmp     byte [esi], 20h
1869 0000817B 7215                <1>     jb      short cmd_echo_nextline
1870                              <1>     ; 14/04/2016
1871 0000817D 56                  <1>     push    esi
1872                              <1> cmd_echo_asciiz:
1873                              <1>     ;inc    esi
1874                              <1>     ;mov    al, [esi]
1875                              <1>     ; 22/11/2017
1876 0000817E AC                  <1>     lodsb
1877 0000817F 3C20                <1>     cmp     al, 20h
1878 00008181 73FB                <1>     jnb     short cmd_echo_asciiz
1879 00008183 4E                  <1>     dec     esi
1880 00008184 C60600              <1>     mov     byte [esi], 0
1881 00008187 5E                  <1>     pop     esi
1882 00008188 89F7                <1>     mov     edi, esi
1883 0000818A E81BECFFFF          <1>     call    print_msg
1884 0000818F C60700              <1>     mov     byte [edi], 0
1885                              <1> cmd_echo_nextline:
1886 00008192 BE[55390100]        <1>     mov     esi, NextLine
1887                              <1>     ;call   print_msg
1888                              <1>     ;retn
1889 00008197 E90EECFFFF          <1>     jmp     print_msg
1890                              <1>
1891                              <1> cmp_cmd_copy:
1892 0000819C B104                <1>     mov     cl, 4
1893 0000819E BF[282F0100]        <1>     mov     edi, Cmd_Copy
1894 000081A3 E8AF020000          <1>     call    cmp_cmd
1895                              <1>     ;jnc    copy_file
1896                              <1>     ; 26/07/2022
1897 000081A8 7205                <1>     jc      short cmp_cmd_move
1898 000081AA E995160000          <1>     jmp     copy_file
1899                              <1>
1900                              <1> cmp_cmd_move:
1901 000081AF B104                <1>     mov     cl, 4
1902 000081B1 BF[2D2F0100]        <1>     mov     edi, Cmd_Move
1903 000081B6 E89C020000          <1>     call    cmp_cmd
1904                              <1>     ;jnc    move_file
1905                              <1>     ; 26/07/2022
1906 000081BB 7205                <1>     jc      short cmp_cmd_path
1907 000081BD E940150000          <1>     jmp     move_file
1908                              <1>
1909                              <1> cmp_cmd_path:
1910 000081C2 B104                <1>     mov     cl, 4
1911 000081C4 BF[322F0100]        <1>     mov     edi, Cmd_Path
1912 000081C9 E889020000          <1>     call    cmp_cmd
1913                              <1>     ;jnc    set_get_path
1914                              <1>     ; 26/07/2022
1915 000081CE 7205                <1>     jc      short cmp_cmd_beep
1916 000081D0 E995180000          <1>     jmp     set_get_path
1917                              <1>
1918                              <1> cmp_cmd_beep:
1919 000081D5 B104                <1>     mov     cl, 4
1920 000081D7 BF[5F2F0100]        <1>     mov     edi, Cmd_Beep
1921 000081DC E876020000          <1>     call    cmp_cmd
1922 000081E1 720B                <1>     jc      short cmp_cmd_find
1923                              <1>     ; 13/05/2016
1924 000081E3 8A3D[AE760100]      <1>     mov     bh, [ptty] ; [ACTIVE_PAGE]
1925 000081E9 E9D7A1FFFF          <1>     jmp     beeper
1926                              <1>
1927                              <1> cmp_cmd_find:
1928 000081EE B104                <1>     mov     cl, 4
1929 000081F0 BF[3C2F0100]        <1>     mov     edi, Cmd_Find
1930 000081F5 E85D020000          <1>     call    cmp_cmd
1931                              <1>         ;jc cmp_cmd_external
1932                              <1>     ; 26/07/2022
1933 000081FA 7305                <1>     jnc     short c4_find
1934 000081FC E939020000          <1>     jmp     cmp_cmd_external
1935                              <1> c4_find:
1936                              <1>     ;call   find_and_list_files
1937 00008201 E9FC200000          <1>     jmp     find_and_list_files
1938                              <1>     ;retn
1939                              <1>
1940                              <1> c_1:
1941 00008206 AD                  <1>     lodsd
1942                              <1> cmp_cmd_help:
1943 00008207 3C3F                <1>     cmp     al, '?'
1944 00008209 751D                <1>         jne     short cmp_cmd_remark
1945                              <1>
1946 0000820B BE[C52E0100]        <1>     mov     esi, Command_List
1947                              <1> cmd_help_next_w:
1948 00008210 E895EBFFFF          <1>     call    print_msg
1949                              <1>
1950 00008215 803E20              <1>     cmp     byte [esi], 20h ; 0
1951 00008218 7233                <1>     jb      short cmd_help_retn
1952                              <1>
1953 0000821A 56                  <1>     push    esi
1954 0000821B BE[E7380100]        <1>     mov     esi, nextline
1955 00008220 E885EBFFFF          <1>     call    print_msg
1956 00008225 5E                  <1>     pop     esi
1957 00008226 EBE8                <1>     jmp     short cmd_help_next_w
1958                              <1>
1959                              <1> cmp_cmd_remark:
1960 00008228 3C2A                <1>     cmp     al, '*'
1961                              <1>     ;jne    cmp_cmd_external
1962                              <1>     ; 26/07/2022
1963 0000822A 7405                <1>     je      short cmp_cmd_rem
1964 0000822C E909020000          <1>     jmp     cmp_cmd_external
1965                              <1> cmp_cmd_rem:
1966 00008231 46                  <1>     inc     esi
```

```
1967 00008232 BF[A4770100]        <1>     mov     edi, Remark
1968 00008237 8A06                <1>     mov     al, [esi]
1969 00008239 3C20                <1>     cmp     al, 20h
1970 0000823B 7707                <1>     ja      short cmd_remark_write
1971 0000823D 89FE                <1>     mov     esi, edi ; Remark
1972 0000823F E966EBFFFF          <1>     jmp     print_msg
1973                              <1>
1974                              <1> cmd_remark_write:
1975 00008244 AA                  <1>     stosb
1976 00008245 AC                  <1>     lodsb
1977 00008246 3C20                <1>     cmp     al, 20h
1978 00008248 73FA                <1>     jnb     short cmd_remark_write
1979 0000824A C60700              <1>     mov     byte [edi], 0
1980                              <1>
1981                              <1> cmd_help_retn:
1982                              <1> cmd_remark_retn:
1983                              <1> cd_retn:
1984 0000824D C3                  <1>     retn
1985                              <1> c_2:
1986                              <1>     ; 26/07/2022
1987 0000824E BE[F2770100]        <1>     mov     esi, CommandBuffer
1988 00008253 80F902              <1>     cmp     cl, 2
1989                              <1>     ;ja     c_3
1990                              <1>     ;mov    esi, CommandBuffer
1991 00008256 72AE                <1>     jb      short c_1
1992                              <1>     ; 26/07/2022
1993                              <1>     ;jne    short c_1
1994                              <1>
1995                              <1> cmp_cmd_cd:
1996 00008258 66AD                <1>     lodsw
1997 0000825A 663D4344            <1>     cmp     ax, 'CD'
1998 0000825E 754E                <1>     jne     short cmp_cmd_drive
1999 00008260 46                  <1>         inc esi
2000                              <1> cd_0:
2001 00008261 668B06              <1>     mov     ax, [esi]
2002 00008264 3C20                <1>     cmp     al, 20h
2003 00008266 76E5                <1>     jna     short cd_retn
2004                              <1>     ; 10/02/2016
2005 00008268 80FC3A              <1>     cmp     ah, ':'
2006 0000826B 7504                <1>     jne     short cd_1
2007 0000826D 46                  <1>     inc     esi
2008 0000826E 46                  <1>     inc     esi
2009 0000826F EB47                <1>     jmp     short cd_2
2010                              <1>
2011                              <1> cd_1:       ; change current directory
2012                              <1>     ; 29/11/2009
2013                              <1>     ; AH = CDh     ; to separate 'CD' command from others
2014                              <1>                    ; for restoring current directory
2015                              <1>                    ; 0CDh sign is for saving cdir into
2016                              <1>                    ; DOS drv description table cdir area
2017                              <1>
2018 00008271 B4CD                <1>     mov     ah, 0CDh ; mov byte [CD_COMMAND], 0CDh
2019                              <1>
2020 00008273 E863210000          <1>     call    change_current_directory
2021                              <1>     ;jnc    change_prompt_dir_string
2022                              <1>     ; 26/07/2022
2023 00008278 7205                <1>     jc      short cd_error_messages
2024 0000827A E985200000          <1>     jmp     change_prompt_dir_string
2025                              <1>
2026                              <1> cd_error_messages:
2027 0000827F 3C03                <1>     cmp     al, 3
2028 00008281 740C                <1>     je      short cd_path_not_found
2029                              <1>     ; 16/10/2016 (15h -> 15)
2030 00008283 3C0F                <1>     cmp     al, 15 ; drive not ready error
2031 00008285 7453                <1>     je      short cd_drive_not_ready
2032 00008287 3C11                <1>     cmp     al, 17 ; read error
2033 00008289 744F                <1>     je      short cd_drive_not_ready
2034 0000828B 3C13                <1>     cmp     al, 19 ; ; Bad directory/path name
2035 0000828D 7460                <1>     je      short cd_command_failed
2036                              <1>
2037                              <1> cd_path_not_found:
2038 0000828F 50                  <1>     push    eax ; 29/12/2017
2039                              <1>     ;push   ax
2040 00008290 BE[68310100]        <1>     mov     esi, Msg_Dir_Not_Found
2041 00008295 E810EBFFFF          <1>     call    print_msg
2042                              <1>     ;pop    ax
2043 0000829A 58                  <1>     pop     eax ; 29/12/2017
2044 0000829B 3A25[40770100]      <1>     cmp     ah, [Current_Dir_Level]
2045                              <1>         ;jnb       change_prompt_dir_string
2046                              <1>     ; 26/07/2022
2047 000082A1 7306                <1>     jnb     short cd_cpds
2048 000082A3 8825[40770100]      <1>     mov     [Current_Dir_Level], ah
2049                              <1> cd_cpds:
2050 000082A9 E956200000          <1>         jmp     change_prompt_dir_string
2051                              <1>
2052                              <1> cmp_cmd_drive: ; change current drive
2053                              <1>     ; C:, D:, E: etc.
2054 000082AE 80FC3A              <1>     cmp     ah, ':'
2055                              <1>     ;jne    cmp_cmd_external
2056                              <1>     ; 26/07/2022
2057 000082B1 7405                <1>     je      short cd_2
2058                              <1> cmd_ext:
2059 000082B3 E982010000          <1>     jmp     cmp_cmd_external
2060                              <1>
2061                              <1> cd_2:       ; 'CD C:', 'CD D:' ...
2062 000082B8 803E20              <1>     cmp     byte [esi], 20h
2063                              <1>     ;ja     loc_cmd_failed
2064                              <1>     ; 26/07/2022
2065 000082BB 7706                <1>     ja      short cd_failed
2066 000082BD 24DF                <1>     and     al, 0DFh
2067 000082BF 2C41                <1>     sub     al, 'A'
2068                              <1>     ;jc     loc_cmd_failed
2069                              <1>     ; 26/07/2022
2070 000082C1 7305                <1>     jnc     short cd_3
2071                              <1> cd_failed:
2072 000082C3 E97C010000          <1>     jmp     loc_cmd_failed
2073                              <1> cd_3:
2074 000082C8 3A05[5C2E0100]      <1>         cmp     al, [Last_DOS_DiskNo]
2075 000082CE 770A                <1>         ja  short cd_drive_not_ready
2076                              <1>
2077 000082D0 88C2                <1>     mov     dl, al
2078 000082D2 E809F4FFFF          <1>     call    change_current_drive
2079 000082D7 7201                <1>     jc      short cd_drive_not_ready
2080 000082D9 C3                  <1>     retn
2081                              <1>
2082                              <1> cd_drive_not_ready:
2083 000082DA BE[25310100]        <1>     mov     esi, Msg_Not_Ready_Read_Err
2084 000082DF E8C6EAFFFF          <1>     call    print_msg
2085                              <1>
2086                              <1> cd_fail_drive_restart:
2087 000082E4 8A15[42770100]      <1>     mov     dl, [Current_Drv]
2088                              <1>     ;call   change_current_drive
2089 000082EA E9F1F3FFFF          <1>         jmp     change_current_drive
2090                              <1>     ;retn
```

```
2091                                    <1>
2092                                    <1> cd_command_failed:
2093 000082EF BE[06310100]             <1>     mov     esi, Msg_Bad_Command
2094 000082F4 E8B1EAFFFF               <1>     call    print_msg
2095 000082F9 EBE9                     <1>     jmp     short cd_fail_drive_restart
2096                                    <1>
2097                                    <1> c_5:
2098                                    <1> cmp_cmd_mkdir:
2099 000082FB BF[222F0100]             <1>     mov     edi, Cmd_Mkdir
2100 00008300 E852010000               <1>     call    cmp_cmd
2101                                    <1>     ;jnc    make_directory
2102                                    <1>     ; 26/07/2022
2103 00008305 7205                     <1>     jc      short cmp_cmd_rmdir
2104 00008307 E9550A0000               <1>     jmp     make_directory
2105                                    <1>
2106                                    <1> cmp_cmd_rmdir:
2107 0000830C B105                     <1>     mov     cl, 5
2108 0000830E BF[1C2F0100]             <1>     mov     edi, Cmd_Rmdir
2109 00008313 E83F010000               <1>     call    cmp_cmd
2110                                    <1>     ;jnc    delete_directory
2111                                    <1>     ; 26/07/2022
2112 00008318 7205                     <1>     jc      short cmp_cmd_chdir
2113 0000831A E95E0B0000               <1>     jmp     delete_directory
2114                                    <1>
2115                                    <1> cmp_cmd_chdir:
2116 0000831F B105                     <1>     mov     cl, 5
2117 00008321 BF[592F0100]             <1>     mov     edi, Cmd_Chdir
2118 00008326 E82C010000               <1>     call    cmp_cmd
2119                                    <1>     ;jc     cmp_cmd_external
2120                                    <1>     ; 26/07/2022
2121 0000832B 7286                     <1>     jc      short cmd_ext
2122                                    <1>
2123 0000832D E92FFFFFFF               <1>     jmp     cd_0
2124                                    <1>
2125                                    <1> c_6:
2126 00008332 80F906                   <1>     cmp     cl, 6
2127                                    <1>     ;ja     c_8
2128                                    <1>     ; 26/07/2022
2129 00008335 72C4                     <1>     jb      short c_5
2130 00008337 7405                     <1>     je      short cmd_6
2131 00008339 E9E4000000               <1>     jmp     c_8
2132                                    <1>
2133                                    <1> cmd_6:
2134                                    <1> cmp_cmd_prompt:
2135 0000833E BF[D82E0100]             <1>     mov     edi, Cmd_Prompt
2136 00008343 E80F010000               <1>     call    cmp_cmd
2137 00008348 722F                     <1>         jc  short cmp_cmd_volume
2138                                    <1> get_prompt_name_fchar:
2139 0000834A AC                       <1>     lodsb
2140 0000834B 3C20                     <1>     cmp     al, 20h
2141 0000834D 74FB                     <1>     je      short get_prompt_name_fchar
2142 0000834F 7713                     <1>     ja      short loc_change_prompt_label
2143                                    <1> default_command_prompt: ; 31/12/2017 ('sysprompt')
2144 00008351 BE[B92E0100]             <1>     mov     esi, TRDOSPromptLabel
2145 00008356 C7065452444F             <1>     mov     dword [esi], "TRDO"
2146 0000835C 66C746045300             <1>             mov     word [esi+4], "S"
2147                                    <1> loc_cmd_prompt_return:
2148 00008362 C3                       <1>     retn
2149                                    <1>
2150                                    <1> set_command_prompt: ; 31/12/2017 ('sysprompt')
2151 00008363 AC                       <1>     lodsb
2152                                    <1> loc_change_prompt_label:
2153                                    <1>     ;mov    cx, 11
2154                                    <1>     ; 26/07/2022
2155 00008364 29C9                     <1>     sub     ecx, ecx
2156 00008366 B10B                     <1>     mov     cl, 11
2157 00008368 BF[B92E0100]             <1>     mov     edi, TRDOSPromptLabel
2158                                    <1> put_char_new_prompt_label:
2159 0000836D AA                       <1>     stosb
2160 0000836E AC                       <1>     lodsb
2161 0000836F 3C20                     <1>     cmp     al, 20h
2162 00008371 7202                     <1>     jb      short pass_put_new_prompt_label
2163 00008373 E2F8                     <1>     loop    put_char_new_prompt_label
2164                                    <1> pass_put_new_prompt_label:
2165 00008375 C60700                   <1>     mov     byte [edi], 0
2166 00008378 C3                       <1>     retn
2167                                    <1>
2168                                    <1> cmp_cmd_volume:
2169 00008379 B106                     <1>     mov     cl, 6
2170 0000837B BF[DF2E0100]             <1>     mov     edi, Cmd_Volume
2171 00008380 E8D2000000               <1>     call    cmp_cmd
2172 00008385 7259                     <1>         jc  short cmp_cmd_attrib
2173                                    <1>
2174                                    <1> cmd_vol1:
2175 00008387 AC                       <1>     lodsb
2176 00008388 3C20                     <1>     cmp     al, 20h
2177 0000838A 7707                     <1>     ja      short cmd_vol2
2178 0000838C A0[42770100]             <1>     mov     al, [Current_Drv]
2179 00008391 EB41                     <1>     jmp     short cmd_vol4
2180                                    <1> cmd_vol2:
2181 00008393 3C41                     <1>     cmp     al, 'A'
2182                                    <1>     ;jb     loc_cmd_failed
2183                                    <1>     ; 26/07/2022
2184 00008395 722D                     <1>     jb      short cmd_vol_failed_1
2185 00008397 3C7A                     <1>     cmp     al, 'z'
2186                                    <1>     ;ja     loc_cmd_failed
2187                                    <1>     ; 26/07/2022
2188 00008399 7731                     <1>     ja      short cmd_vol_failed_2
2189 0000839B 3C5A                     <1>     cmp     al, 'Z'
2190 0000839D 7606                     <1>     jna     short cmd_vol3
2191 0000839F 3C61                     <1>     cmp     al, 'a'
2192                                    <1>         ;jb loc_cmd_failed
2193                                    <1>     ; 26/07/2022
2194 000083A1 722B                     <1>     jb      short cmd_vol_failed_3
2195 000083A3 24DF                     <1>     and     al, 0DFh
2196                                    <1> cmd_vol3:
2197 000083A5 8A26                     <1>     mov     ah, [esi]
2198 000083A7 80FC3A                   <1>     cmp     ah, ':'
2199 000083AA 0F8594000000             <1>         jne     loc_cmd_failed
2200 000083B0 2C41                     <1>     sub     al, 'A'
2201 000083B2 3A05[5C2E0100]           <1>         cmp     al, [Last_DOS_DiskNo]
2202 000083B8 761A                     <1>     jna     short cmd_vol4
2203                                    <1>
2204 000083BA BE[25310100]             <1>     mov     esi, Msg_Not_Ready_Read_Err
2205 000083BF E9E6E9FFFF               <1>     jmp     print_msg
2206                                    <1>
2207                                    <1>     ; 26/07/2022
2208                                    <1>     ; (numeric characters and underscore are allowed)
2209                                    <1> cmd_vol_failed_1:
2210                                    <1>     ; check for numeric characters
2211 000083C4 3C30                     <1>     cmp     al, '0'
2212 000083C6 7204                     <1>     jb      short cmd_vol_failed_2
2213 000083C8 3C39                     <1>     cmp     al, '9'
2214 000083CA 76D9                     <1>     jna     short cmd_vol3
```

```
2215                            <1> cmd_vol_failed_2:
2216 000083CC EB76             <1>     jmp     loc_cmd_failed
2217                            <1> cmd_vol_failed_3:
2218 000083CE 3C5F             <1>     cmp     al, '_' ; underline ?
2219 000083D0 74D3             <1>     je      short cmd_vol3 ; is ok..
2220 000083D2 EBF8             <1>     jmp     short cmd_vol_failed_2
2221                            <1>
2222                            <1> cmd_vol4:
2223 000083D4 E8DEFAFFFF       <1>     call    print_volume_info
2224                            <1>     ;jc     cd_drive_not_ready
2225                            <1>     ; 26/07/2022
2226 000083D9 7339             <1>     jnc     short cmd_vol5
2227 000083DB E9FAFEFFFF       <1>     jmp     cd_drive_not_ready
2228                            <1> ;cmd_vol5:
2229                            <1> ;   retn
2230                            <1>
2231                            <1> cmp_cmd_attrib:
2232 000083E0 B106             <1>     mov     cl, 6
2233 000083E2 BF[0E2F0100]     <1>     mov     edi, Cmd_Attrib
2234 000083E7 E86B000000       <1>     call    cmp_cmd
2235                            <1>     ;jnc    set_file_attributes
2236                            <1>     ; 26/07/2022
2237 000083EC 7205             <1>     jc      short cmp_cmd_rename
2238 000083EE E9D80E0000       <1>     jmp     set_file_attributes
2239                            <1>
2240                            <1> cmp_cmd_rename:
2241 000083F3 B106             <1>     mov     cl, 6
2242 000083F5 BF[152F0100]     <1>     mov     edi, Cmd_Rename
2243 000083FA E858000000       <1>     call    cmp_cmd
2244                            <1>     ;jnc    rename_file
2245                            <1>     ; 26/07/2022
2246 000083FF 7205             <1>     jc      short cmp_cmd_device
2247 00008401 E9BF100000       <1>     jmp     rename_file
2248                            <1>
2249                            <1> cmp_cmd_device:
2250 00008406 B106             <1>     mov     cl, 6
2251 00008408 BF[4A2F0100]     <1>     mov     edi, Cmd_Device
2252 0000840D E845000000       <1>     call    cmp_cmd
2253 00008412 7226             <1>     jc      short cmp_cmd_external
2254                            <1>     ; 26/07/2022
2255                            <1> cmd_vol5:
2256                            <1> cmd_dev:
2257 00008414 C3               <1>     retn
2258                            <1>
2259                            <1> c_7:
2260                            <1> cmp_cmd_devlist:
2261 00008415 BF[512F0100]     <1>     mov     edi, Cmd_DevList
2262 0000841A E838000000       <1>     call    cmp_cmd
2263 0000841F 7219             <1>         jc  short cmp_cmd_external
2264                            <1>
2265                            <1> loc_cmd_return:
2266 00008421 C3               <1>     retn
2267                            <1>
2268                            <1> c_8:
2269 00008422 80F908           <1>         cmp cl, 8
2270 00008425 7713             <1>     ja      short cmp_cmd_external
2271 00008427 72EC             <1>     jb      short c_7
2272                            <1>
2273                            <1> cmp_cmd_longname:
2274 00008429 BF[E62E0100]     <1>     mov     edi, Cmd_LongName
2275 0000842E E824000000       <1>     call    cmp_cmd
2276                            <1>     ;jnc    get_and_print_longname
2277                            <1>     ; 26/07/2022
2278 00008433 7205             <1>     jc      short cmp_cmd_external
2279 00008435 E95A060000       <1>     jmp     get_and_print_longname
2280                            <1>
2281                            <1> cmp_cmd_external:
2282                            <1>     ; 07/05/2016
2283                            <1>     ; 22/04/2016
2284 0000843A BE[F2770100]     <1>     mov     esi, CommandBuffer
2285 0000843F E9A61B0000       <1>     jmp     loc_run_check_filename
2286                            <1>
2287                            <1> loc_cmd_failed:
2288 00008444 803D[F2770100]20 <1>     cmp     byte [CommandBuffer], 20h
2289 0000844B 76D4             <1>     jna     short loc_cmd_return
2290 0000844D BE[06310100]     <1>     mov     esi, Msg_Bad_Command
2291                            <1> ;   call    print_msg
2292                            <1> ;loc_cmd_return:
2293                            <1> ;   retn
2294 00008452 E953E9FFFF       <1>     jmp     print_msg
2295                            <1>
2296                            <1> cmp_cmd:
2297                            <1>     ; 26/07/2022 (TRDOS 386 Kernel v2.0.5)
2298                            <1>     ; 29/01/2016 (TRDOS 386 = TRDOS v2.0)
2299 00008457 BE[F2770100]     <1>         mov esi, CommandBuffer
2300                            <1>         ; edi = internal command word (ASCIIZ)
2301                            <1>     ; ecx = command length (<=8)
2302                            <1> cmp_cmd_1:
2303 0000845C AC               <1>     lodsb
2304 0000845D AE               <1>     scasb
2305 0000845E 750D             <1>     jne     short cmp_cmd_3
2306 00008460 E2FA             <1>     loop    cmp_cmd_1
2307 00008462 AC               <1>     lodsb
2308 00008463 3C20             <1>     cmp     al, 20h
2309 00008465 7703             <1>     ja      short cmp_cmd_2
2310 00008467 30C0             <1>     xor     al, al
2311                            <1>     ; ZF = 1 -> internal command word matches
2312 00008469 C3               <1>     retn
2313                            <1> cmp_cmd_2:
2314                            <1>     ; ZF = 0 (CF = 0) -> external command word
2315 0000846A 58               <1>     pop     eax ; no return to the caller from here
2316 0000846B EBCD             <1>     jmp     short cmp_cmd_external ; 26/07/2022
2317                            <1> cmp_cmd_3:
2318 0000846D F9               <1>     stc
2319                            <1>     ; CF = 1 -> internal command word does not match
2320 0000846E C3               <1>     retn
2321                            <1>
2322                            <1> loc_run_cmd_failed:
2323                            <1>     ; 26/07/2022 (TRDOS 386 Kernel v2.0.5)
2324                            <1>     ; 15/03/2016
2325                            <1>     ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
2326                            <1>     ; 07/12/2009 (CMD_INTR.ASM)
2327                            <1>     ; 29/11/2009
2328                            <1>
2329 0000846F E863000000       <1>     call    restore_cdir_after_cmd_fail
2330                            <1>
2331                            <1> loc_run_cmd_failed_cmp_al:
2332                            <1>     ; End of Restore_CDIR code (29/11/2009)
2333                            <1>
2334 00008474 3C01             <1>     cmp     al, 1 ; Bad command or file name
2335 00008476 74CC             <1>     je      short loc_cmd_failed ; 26/07/2022
2336                            <1> loc_run_dir_not_found:
2337 00008478 3C03             <1>     cmp     al, 3
2338 0000847A 750A             <1>     jne     short loc_run_file_notfound_msg
```

```
2339                                    <1>     ; Path not found (MS-DOS Error Code = 3)
2340 0000847C BE[68310100]              <1>     mov     esi, Msg_Dir_Not_Found
2341 00008481 E924E9FFFF                <1>     jmp     print_msg
2342                                    <1>
2343                                    <1> loc_run_file_notfound_msg:
2344 00008486 3C02                      <1>     cmp     al, 2 ; File not found
2345 00008488 750A                      <1>     jne     short loc_run_file_drv_read_err
2346                                    <1>
2347                                    <1> loc_print_file_notfound_msg:
2348 0000848A BE[7F310100]              <1>         mov     esi, Msg_File_Not_Found
2349                                    <1>     ;call   proc_printmsg
2350                                    <1>     ;retn
2351 0000848F E916E9FFFF                <1>     jmp     print_msg
2352                                    <1>
2353                                    <1> loc_run_file_drv_read_err:
2354                                    <1>     ; Err: 17 (Read fault)
2355 00008494 3C11                      <1>     cmp     al, 17 ; Drive not ready or read error
2356 00008496 7404                      <1>     je      short loc_run_file_print_drv_read_err
2357                                    <1>     ;
2358 00008498 3C0F                      <1>     cmp     al, 15 ; Drive not ready (or read error)
2359 0000849A 750A                      <1>     jne     short loc_run_file_toobig
2360                                    <1>
2361                                    <1> loc_run_file_print_drv_read_err:
2362 0000849C BE[25310100]              <1>     mov     esi, Msg_Not_Ready_Read_Err
2363 000084A1 E904E9FFFF                <1>     jmp     print_msg
2364                                    <1>
2365                                    <1> loc_run_file_toobig:
2366 000084A6 3C08                      <1>     cmp     al, 8 ; Not enough free memory to load&run file
2367 000084A8 750A                      <1>     jne     short loc_run_file_perm_denied
2368 000084AA BE[CA310100]              <1>     mov     esi, Msg_Insufficient_Memory
2369 000084AF E9F6E8FFFF                <1>     jmp     print_msg
2370                                    <1>
2371                                    <1> loc_run_file_perm_denied:
2372                                    <1>     ; 29/12/2017
2373 000084B4 3C0B                      <1>     cmp     al, ERR_PERM_DENIED ; 11 ; Permission denied
2374 000084B6 750A                      <1>     jne     short loc_run_misc_error
2375 000084B8 BE[5E330100]              <1>     mov     esi, Msg_Permission_Denied
2376 000084BD E9E8E8FFFF                <1>     jmp     print_msg
2377                                    <1>
2378                                    <1>     ; 15/03/2016
2379                                    <1> print_misc_error_msg:
2380                                    <1> loc_run_misc_error:
2381                                    <1>     ; AL = Error code
2382 000084C2 E8F4BCFFFF                <1>     call    bytetohex
2383 000084C7 66A3[FE310100]            <1>         mov     [error_code_hex], ax
2384                                    <1>
2385 000084CD BE[E1310100]              <1>     mov     esi, Msg_Error_Code
2386                                    <1>     ;call   print_msg
2387                                    <1>     ;retn
2388 000084D2 E9D3E8FFFF                <1>     jmp     print_msg
2389                                    <1>
2390                                    <1> restore_cdir_after_cmd_fail:
2391                                    <1>     ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
2392 000084D7 50                        <1>     push    eax
2393 000084D8 8A3D[9F7E0100]            <1>     mov     bh, [RUN_CDRV] ; it is set at the beginning
2394                                    <1>                            ; of the 'run' command.
2395 000084DE 3A3D[42770100]            <1>     cmp     bh, [Current_Drv]
2396 000084E4 7409                      <1>     je      short loc_run_restore_cdir
2397 000084E6 88FA                      <1>     mov     dl, bh
2398 000084E8 E8F3F1FFFF                <1>     call    change_current_drive
2399 000084ED EB19                      <1>     jmp     short loc_run_err_pass_restore_cdir
2400                                    <1>
2401                                    <1> loc_run_restore_cdir:
2402 000084EF 803D[5D2E0100]00          <1>     cmp     byte [Restore_CDIR], 0
2403 000084F6 7610                      <1>     jna     short loc_run_err_pass_restore_cdir
2404 000084F8 30DB                      <1>     xor     bl, bl
2405 000084FA 0FB7F3                    <1>     movzx   esi, bx
2406 000084FD 81C600010900              <1>     add     esi, Logical_DOSDisks
2407 00008503 E889F2FFFF                <1>     call    restore_current_directory
2408                                    <1>
2409                                    <1> loc_run_err_pass_restore_cdir:
2410 00008508 58                        <1>     pop     eax
2411 00008509 C3                        <1>     retn
2412                                    <1>
2413                                    <1> print_directory_list:
2414                                    <1>     ; 02/12/2023 (TRDOS 386 v2.0.7)
2415                                    <1>     ; 07/08/2022
2416                                    <1>     ; 27/07/2022 (TRDOS 386 Kernel v2.0.5)
2417                                    <1>     ; 10/02/2016
2418                                    <1>     ; 08/02/2016 (TRDOS 386 = TRDOS v2.0)
2419                                    <1>     ; 06/12/2009 ('cmp_cmd_dir')
2420                                    <1>     ;
2421 0000850A 66C705[E07F0100]00-         <1>     mov     word [AttributesMask], 0800h ; ..except volume names..
2421 00008512 08                        <1>
2422 00008513 A0[42770100]              <1>     mov     al, [Current_Drv]
2423 00008518 A2[9F7E0100]              <1>     mov     [RUN_CDRV], al
2424                                    <1>     ; 02/12/2023
2425 0000851D 3C02                      <1>     cmp     al, 2
2426 0000851F 733A                      <1>     jnb     short get_dfname_fchar
2427 00008521 A0[01770100]              <1>     mov     al, [TIMER_LOW+1]
2428 00008526 D0E8                      <1>     shr     al, 1   ; 512/18.2 (>= 28 seconds)
2429 00008528 8605[667E0100]            <1>     xchg    al, [P_TIMER]  ; 28 seconds
2430 0000852E 3A05[667E0100]            <1>     cmp     al, [P_TIMER]
2431 00008534 7425                      <1>     je      short get_dfname_fchar
2432 00008536 E893F1FFFF                <1>     call    get_media_change_status
2433 0000853B 7205                      <1>     jc      short pdl_chdrv
2434 0000853D 80FC06                    <1>     cmp     ah, 6
2435 00008540 7519                      <1>     jne     short get_dfname_fchar
2436                                    <1> pdl_chdrv:
2437 00008542 56                        <1>     push    esi
2438 00008543 8A15[9F7E0100]            <1>     mov     dl, [RUN_CDRV]
2439 00008549 E892F1FFFF                <1>     call    change_current_drive
2440 0000854E 5E                        <1>     pop     esi
2441 0000854F 730A                      <1>     jnc     short get_dfname_fchar
2442 00008551 BE[25310100]              <1>     mov     esi, Msg_Not_Ready_Read_Err
2443                                    <1>     ;call   print_msg
2444                                    <1>     ;retn
2445 00008556 E94FE8FFFF                <1>     jmp     print_msg
2446                                    <1>
2447                                    <1> get_dfname_fchar:
2448 0000855B AC                        <1>     lodsb
2449 0000855C 3C20                      <1>     cmp     al, 20h
2450 0000855E 74FB                      <1>     je      short get_dfname_fchar
2451                                    <1>     ;jb     loc_print_dir_call_all
2452                                    <1>     ; 02/12/2023
2453 00008560 7705                      <1>     ja      short get_dfname_fchar_2
2454 00008562 E9E0000000                <1>     jmp     loc_print_dir_call_all
2455                                    <1> get_dfname_fchar_2:
2456 00008567 3C2D                      <1>     cmp     al, '-'
2457 00008569 753C                      <1>     jne     short loc_print_dir_call_flt
2458                                    <1> get_next_attr_char:
2459 0000856B AC                        <1>     lodsb
2460 0000856C 3C20                      <1>     cmp     al, 20h
2461 0000856E 74FB                      <1>     je      short get_next_attr_char
```

```
                                <1>     ;jb    loc_cmd_failed
                                <1>     ; 27/07/2022
00008570 7305                   <1>     jnb    short pdl_1
                                <1> pdl_0:
00008572 E9CDFEFFFF             <1>     jmp    loc_cmd_failed
                                <1> pdl_1:
00008577 24DF                   <1>     and    al, 0DFh
00008579 3C44                   <1>     cmp    al, 'D' ; directories only ?
0000857B 750E                   <1>     jne    short pass_only_directories
0000857D AC                     <1>     lodsb
0000857E 3C20                   <1>     cmp    al, 20h
                                <1>     ;ja    loc_cmd_failed
                                <1>     ; 27/07/2022
00008580 77F0                   <1>     ja     short pdl_0
00008582 800D[E07F0100]10       <1>     or     byte [AttributesMask], 10h ; ..directory..
00008589 EB10                   <1>     jmp    short get_dfname_fchar_attr
                                <1> pass_only_directories:
0000858B 3C46                   <1>     cmp    al, 'F' ; files only ?
0000858D 7530                   <1>     jne    short check_attr_s ; 27/07/2022
0000858F AC                     <1>     lodsb
00008590 3C20                   <1>     cmp    al, 20h
                                <1>     ;ja    loc_cmd_failed
                                <1>     ; 27/07/2022
00008592 77DE                   <1>     ja     short pdl_0
00008594 800D[E17F0100]10       <1>     or     byte [AttributesMask+1], 10h ; ..except directories..
                                <1> get_dfname_fchar_attr:
0000859B AC                     <1>     lodsb
0000859C 3C20                   <1>     cmp    al, 20h
0000859E 74FB                   <1>     je     short get_dfname_fchar_attr
                                <1>     ;jb    short loc_print_dir_call_all
                                <1>     ; 07/08/2022
000085A0 7305                   <1>     jnb    short loc_print_dir_call_flt
000085A2 E9A0000000             <1>     jmp    loc_print_dir_call_all
                                <1> loc_print_dir_call_flt:
000085A7 4E                     <1>     dec    esi
000085A8 BF[E27F0100]           <1>     mov    edi, FindFile_Drv
000085AD E8BC230000             <1>     call   parse_path_name
000085B2 7352                   <1>     jnc    short loc_print_dir_change_drv_1
000085B4 3C01                   <1>     cmp    al, 1
                                <1>     ;ja    loc_run_cmd_failed
                                <1>     ; 27/07/2022
000085B6 764E                   <1>     jna    short loc_print_dir_change_drv_1
                                <1> pdl_2:
000085B8 E9B2FEFFFF             <1>     jmp    loc_run_cmd_failed
                                <1>
                                <1>     ; 27/07/2022
                                <1> check_attr_s_cap:
000085BD 24DF                   <1>     and    al, 0DFh
                                <1> check_attr_s:
000085BF 3C53                   <1>     cmp    al, 'S'
000085C1 7510                   <1>     jne    short pass_attr_s
000085C3 800D[E07F0100]04       <1>     or     byte [AttributesMask], 4 ; system
000085CA AC                     <1>     lodsb
000085CB 3C20                   <1>     cmp    al, 20h
000085CD 74CC                   <1>     je     short get_dfname_fchar_attr
000085CF 7276                   <1>     jb     short loc_print_dir_call_all
000085D1 24DF                   <1>     and    al, 0DFh
                                <1> pass_attr_s:
000085D3 3C48                   <1>     cmp    al, 'H'
000085D5 7510                   <1>     jne    short pass_attr_h
000085D7 800D[E07F0100]02       <1>     or     byte [AttributesMask], 2 ; hidden
                                <1> pass_attr_shr:
000085DE AC                     <1>     lodsb
000085DF 3C20                   <1>     cmp    al, 20h
000085E1 74B8                   <1>     je     short get_dfname_fchar_attr
000085E3 7262                   <1>     jb     short loc_print_dir_call_all
000085E5 EBD6                   <1>     jmp    short check_attr_s_cap
                                <1> pass_attr_h:
000085E7 3C52                   <1>     cmp    al, 'R'
000085E9 7509                   <1>     jne    short pass_attr_r
000085EB 800D[E07F0100]01       <1>     or     byte [AttributesMask], 1 ; read only
000085F2 EBEA                   <1>     jmp    short pass_attr_shr
                                <1> pass_attr_r:
000085F4 3C41                   <1>     cmp    al, 'A'
                                <1>     ;jne   loc_cmd_failed
                                <1>     ; 27/07/2022
000085F6 7405                   <1>     je     short pass_attr_a
000085F8 E947FEFFFF             <1>     jmp    loc_cmd_failed
                                <1> pass_attr_a:
000085FD 800D[E07F0100]20       <1>     or     byte [AttributesMask], 20h ; archive
00008604 EBD8                   <1>     jmp    short pass_attr_shr
                                <1>
                                <1>     ; 07/08/2022
                                <1> loc_print_dir_change_drv_1:
00008606 8A15[E27F0100]         <1>     mov    dl, [FindFile_Drv]
                                <1> loc_print_dir_change_drv_2:
0000860C 3A15[9F7E0100]         <1>     cmp    dl, [RUN_CDRV]
00008612 7407                   <1>     je     short loc_print_dir_change_directory
00008614 E8C7F0FFFF             <1>     call   change_current_drive
                                <1>         ;jc loc_run_cmd_failed
                                <1>     ; 27/07/2022
00008619 729D                   <1>     jc     short pdl_2
                                <1> loc_print_dir_change_directory:
0000861B 803D[E37F0100]20       <1>     cmp    byte [FindFile_Directory], 20h ; 0 or 20h ?
00008622 7619                   <1>     jna    short pass_print_dir_change_directory
                                <1>
00008624 FE05[5D2E0100]         <1>     inc    byte [Restore_CDIR]
0000862A BE[E37F0100]           <1>     mov    esi, FindFile_Directory
0000862F 30E4                   <1>     xor    ah, ah ; CD_COMMAND sign -> 0
00008631 E8A51D0000             <1>     call   change_current_directory
                                <1>         ;jc loc_run_cmd_failed
                                <1>     ; 27/07/2022
00008636 7280                   <1>     jc     short pdl_2
                                <1>
                                <1> loc_print_dir_change_prompt_dir_string:
00008638 E8C71C0000             <1>     call   change_prompt_dir_string
                                <1>
                                <1> pass_print_dir_change_directory:
0000863D BE[24800100]           <1>     mov    esi, FindFile_Name
00008642 803E20                 <1>     cmp    byte [esi], 20h ; ; 0 or 20h ?
00008645 7706                   <1>     ja     short loc_print_dir_call
                                <1>
                                <1> loc_print_dir_call_all:
00008647 C7062A2E2A00           <1>     mov    dword [esi], '*.*'
                                <1> loc_print_dir_call:
0000864D E82D000000             <1>     call   print_directory
                                <1>
00008652 8A15[9F7E0100]         <1>     mov    dl, [RUN_CDRV]  ; it is set at the beginning
00008658 3A15[42770100]         <1>     cmp    dl, [Current_Drv]
0000865E 7405                   <1>     je     short loc_print_dir_call_restore_cdir_retn
                                <1>     ;call  change_current_drive
                                <1>     ;retn
                                <1>     ; 27/07/2022
00008660 E97BF0FFFF             <1>     jmp    change_current_drive
```

```
2586                                <1>
2587                                <1> loc_print_dir_call_restore_cdir_retn:
2588 00008665 803D[5D2E0100]00     <1>     cmp     byte [Restore_CDIR], 0
2589 0000866C 7610                 <1>     jna     short pass_print_dir_call_restore_cdir_retn
2590                                <1>
2591 0000866E BE00010900           <1>     mov     esi, Logical_DOSDisks
2592 00008673 31C0                 <1>     xor     eax, eax
2593 00008675 88D4                 <1>     mov     ah, dl
2594 00008677 01C6                 <1>     add     esi, eax
2595                                <1>
2596                                <1>     ;call   restore_current_directory
2597                                <1>     ; 27/07/2022
2598 00008679 E913F1FFFF           <1>     jmp     restore_current_directory
2599                                <1>
2600                                <1> pass_print_dir_call_restore_cdir_retn:
2601 0000867E C3                   <1>     retn
2602                                <1>
2603                                <1> print_directory:
2604                                <1>     ; 27/07/2022 (TRDOS 386 Kernel v2.0.5)
2605                                <1>     ; 13/05/2016
2606                                <1>     ; 11/02/2016
2607                                <1>     ; 10/02/2016
2608                                <1>     ; 08/02/2016 (TRDOS 386 = TRDOS v2.0)
2609                                <1>     ; 30/10/2010 ('proc_print_directory')
2610                                <1>     ; 19/09/2009
2611                                <1>     ; 2005
2612                                <1>     ; INPUT ->
2613                                <1>     ;       ESI = Asciiz File/Dir Name Address
2614                                <1>
2615 0000867F 56                   <1>     push    esi
2616                                <1>
2617 00008680 29C0                 <1>     sub     eax, eax
2618                                <1>
2619 00008682 66A3[6C800100]       <1>     mov     [Dir_Count], ax ; 0
2620 00008688 66A3[6A800100]       <1>     mov     [File_Count], ax ; 0
2621 0000868E A3[6E800100]         <1>     mov     [Total_FSize], eax ; 0
2622                                <1>
2623 00008693 E828E7FFFF           <1>     call    clear_screen
2624                                <1>
2625 00008698 31C9                 <1>     xor     ecx, ecx
2626 0000869A 8A2D[42770100]       <1>     mov     ch, [Current_Drv] ; DirBuff_Drv - 'A'
2627 000086A0 A0[43770100]         <1>     mov     al, [Current_Dir_Drv]
2628 000086A5 A2[23300100]         <1>     mov     [Dir_Drive_Name], al
2629 000086AA BE00010900           <1>     mov     esi, Logical_DOSDisks
2630 000086AF 01CE                 <1>     add     esi, ecx
2631                                <1>
2632 000086B1 E856F9FFFF           <1>     call    move_volume_name_and_serial_no
2633 000086B6 730C                 <1>     jnc     short print_dir_strlen_check
2634                                <1>
2635 000086B8 5E                   <1>     pop     esi
2636 000086B9 8A3D[AE760100]       <1>     mov     bh, [ptty] ; [ACTIVE_PAGE]
2637                                <1>     ;call   beeper
2638                                <1>     ;retn
2639 000086BF E9019DFFFF           <1>     jmp     beeper  ; beep ! and return
2640                                <1>
2641                                <1> print_dir_strlen_check:
2642 000086C4 BE[45770100]         <1>     mov     esi, Current_Dir_Root
2643 000086C9 BF[C0300100]         <1>     mov     edi, Dir_Str_Root
2644                                <1>
2645                                <1>     ;xor    ecx, ecx
2646 000086CE 8A0D[A1770100]       <1>         mov     cl, [Current_Dir_StrLen]
2647 000086D4 FEC1                 <1>     inc     cl
2648 000086D6 80F940               <1>     cmp     cl, 64
2649 000086D9 760D                 <1>     jna     short pass_print_dir_strlen_shorting
2650 000086DB 46                   <1>     inc     esi
2651 000086DC 01CE                 <1>     add     esi, ecx
2652 000086DE 83EE40               <1>     sub     esi, 64
2653 000086E1 47                   <1>     inc     edi
2654 000086E2 B82E2E2E20           <1>     mov     eax, '... '
2655 000086E7 AB                   <1>     stosd
2656                                <1>
2657                                <1> pass_print_dir_strlen_shorting:
2658 000086E8 F3A4                 <1>     rep     movsb
2659                                <1>
2660 000086EA BE[16300100]         <1>     mov     esi, Dir_Drive_Str
2661 000086EF E8B6E6FFFF           <1>     call    print_msg
2662                                <1>
2663 000086F4 BE[75300100]         <1>     mov     esi, Vol_Serial_Header
2664 000086F9 E8ACE6FFFF           <1>     call    print_msg
2665                                <1>
2666 000086FE BE[B5300100]         <1>     mov     esi, Dir_Str_Header
2667 00008703 E8A2E6FFFF           <1>     call    print_msg
2668                                <1>
2669 00008708 BE[E5380100]         <1>     mov     esi, next2line
2670 0000870D E898E6FFFF           <1>     call    print_msg
2671                                <1>
2672                                <1> loc_print_dir_first_file:
2673 00008712 C605[81800100]10     <1>     mov     byte [PrintDir_RowCounter], 16
2674 00008719 66A1[E07F0100]       <1>     mov     ax, [AttributesMask]
2675 0000871F 5E                   <1>     pop     esi
2676                                <1>
2677 00008720 E845020000           <1>     call    find_first_file
2678                                <1>     ;jc     loc_dir_ok
2679                                <1>     ; 27/07/2022
2680 00008725 7305                 <1>     jnc     short loc_dfname_use_this
2681 00008727 E97B010000           <1>     jmp     loc_dir_ok
2682                                <1>
2683                                <1> loc_dfname_use_this:
2684                                <1>     ; bl =  File Attributes (bh = Long Name Entry Length)
2685 0000872C F6C310               <1>     test    bl, 10h  ; Is it a directory?
2686 0000872F 741B                 <1>     jz      short loc_not_dir
2687                                <1>
2688 00008731 66FF05[6C800100]     <1>     inc     word [Dir_Count]
2689 00008738 89F2                 <1>     mov     edx, esi        ; FindFile_DirEntry address
2690 0000873A BE[04320100]         <1>     mov     esi, Type_Dir  ; '<DIR>      '
2691 0000873F BF[1B320100]         <1>     mov     edi, Dir_Or_FileSize
2692                                <1>     ; move 10 bytes
2693 00008744 A5                   <1>     movsd
2694 00008745 A5                   <1>     movsd
2695 00008746 66A5                 <1>     movsw
2696 00008748 89D6                 <1>     mov     esi, edx
2697 0000874A EB36                 <1>     jmp     short loc_dir_attribute
2698                                <1>
2699                                <1> loc_not_dir:
2700 0000874C 66FF05[6A800100]     <1>     inc     word [File_Count]
2701 00008753 0105[6E800100]       <1>     add     [Total_FSize], eax
2702                                <1>
2703 00008759 B90A000000           <1>     mov     ecx, 10  ; 32 bit divisor
2704 0000875E 89CF                 <1>     mov     edi, ecx
2705 00008760 81C7[1B320100]       <1>     add     edi, Dir_Or_FileSize
2706                                <1> loc_dir_rdivide:
2707 00008766 29D2                 <1>     sub     edx, edx
2708 00008768 F7F1                 <1>     div     ecx     ; remainder in dl (< 10)
2709 0000876A 80C230               <1>     add     dl, '0'         ; to make visible (ascii)
```

```
2710 0000876D 4F                <1>     dec     edi
2711 0000876E 8817              <1>     mov     [edi], dl
2712 00008770 21C0              <1>     and     eax, eax
2713 00008772 75F2              <1>     jnz     short loc_dir_rdivide
2714                            <1>
2715                            <1> loc_dir_fill_space:
2716 00008774 81FF[1B320100]    <1>     cmp     edi, Dir_Or_FileSize
2717 0000877A 7606              <1>     jna     short loc_dir_attribute
2718 0000877C 4F                <1>     dec     edi
2719 0000877D C60720            <1>     mov     byte [edi], 20h
2720 00008780 EBF2              <1>     jmp     short loc_dir_fill_space
2721                            <1>
2722                            <1> loc_dir_attribute:
2723 00008782 C705[26320100]2020- <1>     mov     dword [File_Attribute], 20202020h
2723 0000878A 2020              <1>
2724                            <1>
2725 0000878C 80FB20            <1>     cmp     bl, 20h  ; Is it an archive file?
2726 0000878F 7207              <1>     jb      short loc_dir_pass_arch
2727 00008791 C605[29320100]41  <1>     mov     byte [File_Attribute+3], 'A'
2728                            <1>
2729                            <1> loc_dir_pass_arch:
2730 00008798 80E307            <1>     and     bl, 7
2731 0000879B 7428              <1>     jz      short loc_dir_file_name
2732 0000879D 88DF              <1>     mov     bh, bl
2733 0000879F 80E303            <1>     and     bl, 3
2734 000087A2 38DF              <1>     cmp     bh, bl
2735 000087A4 7607              <1>     jna     short loc_dir_pass_s
2736 000087A6 C605[26320100]53  <1>     mov     byte [File_Attribute], 'S'
2737                            <1>
2738                            <1> loc_dir_pass_s:
2739 000087AD 80E302            <1>     and     bl,2
2740 000087B0 7407              <1>     jz      short loc_dir_pass_h
2741 000087B2 C605[27320100]48  <1>     mov     byte [File_Attribute+1], 'H'
2742                            <1> loc_dir_pass_h:
2743 000087B9 80E701            <1>     and     bh,1
2744 000087BC 7407              <1>     jz      short loc_dir_file_name
2745 000087BE C605[28320100]52  <1>     mov     byte [File_Attribute+2], 'R'
2746                            <1> loc_dir_file_name:
2747                            <1>     ;mov    bx, [esi+18h] ; Date
2748                            <1>     ;mov    dx, [esi+16h] ; Time
2749 000087C5 8B5E16            <1>     mov     ebx, [esi+16h]
2750 000087C8 89F1              <1>     mov     ecx, esi ; FindFile_DirEntry address
2751 000087CA BF[0E320100]      <1>     mov     edi, File_Name
2752                            <1>     ; move 8 bytes
2753 000087CF A5                <1>     movsd
2754 000087D0 A5                <1>     movsd
2755 000087D1 C60720            <1>     mov     byte [edi], 20h
2756 000087D4 47                <1>     inc     edi
2757                            <1>     ; move 3 bytes
2758 000087D5 66A5              <1>     movsw
2759 000087D7 A4                <1>     movsb
2760 000087D8 89CE              <1>     mov     esi, ecx
2761                            <1>
2762                            <1> Dir_Time_start:
2763                            <1>     ;mov    ax, dx          ; Time
2764 000087DA 6689D8            <1>     mov     ax, bx
2765 000087DD 66C1E805          <1>     shr     ax, 5           ; shift right 5 times
2766 000087E1 6683E03F          <1>     and     ax, 0000111111b       ; Minute Mask
2767 000087E5 D40A              <1>     aam                     ; Q([AL]/10)->AH
2768                            <1>                             ; R([AL]/10)->AL
2769                            <1>                             ; [AL]+[AH]= Minute as BCD
2770 000087E7 660D3030          <1>     or      ax, '00'        ; Convert to ASCII
2771 000087EB 86C4              <1>     xchg    ah, al
2772 000087ED 66A3[39320100]    <1>     mov     [File_Minute], ax
2773                            <1>
2774                            <1>     ;mov    al, dh
2775 000087F3 88F8              <1>     mov     al, bh
2776 000087F5 C0E803            <1>     shr     al, 3           ; shift right 3 times
2777 000087F8 D40A              <1>     aam                     ; [AL]+[AH]= Hours as BCD
2778 000087FA 660D3030          <1>     or      ax, '00'
2779 000087FE 86C4              <1>     xchg    ah, al
2780 00008800 66A3[36320100]    <1>     mov     [File_Hour], ax
2781                            <1>
2782 00008806 C1EB10            <1>     shr     ebx, 16         ; BX = Date
2783                            <1>
2784                            <1> Dir_Date_start:
2785                            <1>     ;mov    ax, bx          ; Date
2786                            <1>     ; 27/07/2022
2787 00008809 89D8              <1>     mov     eax, ebx
2788 0000880B 6683E01F          <1>     and     ax, 00011111b  ; Day Mask
2789 0000880F D40A              <1>     aam                     ; Q([AL]/10)->AH
2790                            <1>                             ; R([AL]/10)->AL
2791                            <1>                             ; [AL]+[AH]= Day as BCD
2792 00008811 660D3030          <1>     or      ax, '00'        ; Convert to ASCII
2793 00008815 86E0              <1>     xchg    al, ah
2794                            <1>
2795 00008817 66A3[2B320100]    <1>     mov     [File_Day], ax
2796                            <1>
2797                            <1>     ;mov    ax, bx
2798                            <1>     ; 27/07/2022
2799 0000881D 89D8              <1>     mov     eax, ebx
2800                            <1>     ;shr    ax, 5           ; shift right 5 times
2801 0000881F C1E805            <1>     shr     eax, 5
2802 00008822 6683E00F          <1>     and     ax, 00001111b  ; Month Mask
2803 00008826 D40A              <1>     aam
2804 00008828 660D3030          <1>     or      ax, '00'
2805 0000882C 86C4              <1>     xchg    ah, al
2806 0000882E 66A3[2E320100]    <1>     mov     [File_Month], ax
2807                            <1>
2808                            <1>     ;mov    ax, bx
2809                            <1>     ; 27/07/2022
2810 00008834 89D8              <1>     mov     eax, ebx
2811                            <1>     ;shr    ax, 9
2812 00008836 C1E809            <1>     shr     eax, 9
2813 00008839 6683E07F          <1>     and     ax, 01111111b  ; Result = Year - 1980
2814 0000883D 6605BC07          <1>     add     ax, 1980
2815                            <1>
2816 00008841 B10A              <1>     mov     cl, 10
2817 00008843 F6F1              <1>     div     cl              ; Q -> AL, R -> AH
2818 00008845 80CC30            <1>     or      ah, '0'
2819 00008848 8825[34320100]    <1>     mov     [File_Year+3], ah
2820 0000884E D40A              <1>     aam
2821 00008850 86C4              <1>     xchg    ah, al
2822 00008852 80CC30            <1>     or      ah, '0'   ; Convert to ASCII
2823 00008855 8825[33320100]    <1>     mov     [File_Year+2], ah
2824 0000885B D40A              <1>     aam
2825 0000885D 86E0              <1>     xchg    al, ah
2826 0000885F 660D3030          <1>     or      ax, '00'
2827 00008863 66A3[31320100]    <1>     mov     [File_Year], ax
2828                            <1>
2829                            <1> loc_show_line:
2830 00008869 56                <1>     push    esi
2831 0000886A BE[0E320100]      <1>     mov     esi, File_Name
2832 0000886F E836E5FFFF        <1>     call    print_msg
```

```
2833 00008874 BE[E7380100]        <1>     mov     esi, nextline
2834 00008879 E82CE5FFFF          <1>     call    print_msg
2835 0000887E 5E                  <1>     pop     esi
2836                              <1>
2837 0000887F FE0D[81800100]      <1>     dec     byte [PrintDir_RowCounter]
2838 00008885 740C                <1>     jz      short pause_dir_scroll ; 27/07/2022
2839                              <1>
2840                              <1> loc_next_entry:
2841 00008887 E88B010000          <1>     call    find_next_file
2842                              <1>     ;jnc    loc_dfname_use_this
2843                              <1>     ; 27/07/2022
2844 0000888C 7219                <1>     jc      short loc_dir_ok
2845 0000888E E999FEFFFF          <1>     jmp     loc_dfname_use_this
2846                              <1>
2847                              <1>     ; 27/07/2022
2848                              <1> pause_dir_scroll:
2849 00008893 28E4                <1>     sub     ah, ah
2850 00008895 E88286FFFF          <1>     call    int16h
2851 0000889A 3C1B                <1>     cmp     al, 1Bh
2852 0000889C 7409                <1>     je      short loc_dir_ok
2853 0000889E C605[81800100]10    <1>     mov     byte [PrintDir_RowCounter], 16 ; Reset counter
2854 000088A5 EBE0                <1>         jmp     short loc_next_entry
2855                              <1>
2856                              <1> loc_dir_ok:
2857 000088A7 B90A000000          <1>     mov     ecx, 10
2858                              <1>     ;mov    ax, [Dir_Count]
2859                              <1>     ; 27/07/2022
2860 000088AC 0FB705[6C800100]    <1>     movzx   eax, word [Dir_Count]
2861 000088B3 BF[4F320100]        <1>     mov     edi, Decimal_Dir_Count
2862                              <1>     ;cmp    ax, cx ; 10
2863                              <1>     ; 27/07/2022
2864 000088B8 39C8                <1>     cmp     eax, ecx
2865 000088BA 7216                <1>     jb      short pass_ddc
2866 000088BC 47                  <1>     inc     edi
2867 000088BD 6683F864            <1>     cmp     ax, 100
2868 000088C1 720F                <1>     jb      short pass_ddc
2869 000088C3 47                  <1>     inc     edi
2870 000088C4 663DE803            <1>     cmp     ax, 1000
2871 000088C8 7208                <1>     jb      short pass_ddc
2872 000088CA 47                  <1>     inc     edi
2873 000088CB 663D1027            <1>     cmp     ax, 10000
2874 000088CF 7201                <1>     jb      short pass_ddc
2875 000088D1 47                  <1>     inc     edi
2876                              <1> pass_ddc:
2877 000088D2 886F01              <1>     mov     [edi+1], ch ; 0
2878                              <1> loc_ddc_rediv:
2879 000088D5 31D2                <1>     xor     edx, edx
2880                              <1>     ;div    cx  ; 10
2881                              <1>     ; 27/07/2022
2882 000088D7 F7F1                <1>     div     ecx
2883 000088D9 80C230              <1>     add     dl, '0'
2884 000088DC 8817                <1>     mov     [edi], dl
2885 000088DE 4F                  <1>     dec     edi
2886                              <1>     ;or     ax, ax
2887                              <1>     ; 27/07/2022
2888 000088DF 09C0                <1>     or      eax, eax
2889 000088E1 75F2                <1>     jnz     short loc_ddc_rediv
2890                              <1>
2891 000088E3 66A1[6A800100]      <1>     mov     ax, [File_Count]
2892 000088E9 BF[3E320100]        <1>     mov     edi, Decimal_File_Count
2893                              <1>     ;cmp    ax, cx ; 10
2894                              <1>     ; 27/07/2022
2895 000088EE 39C8                <1>     cmp     eax, ecx ; 10
2896 000088F0 7216                <1>     jb      short pass_dfc
2897 000088F2 47                  <1>     inc     edi
2898 000088F3 6683F864            <1>     cmp     ax, 100
2899 000088F7 720F                <1>     jb      short pass_dfc
2900 000088F9 47                  <1>     inc     edi
2901 000088FA 663DE803            <1>     cmp     ax, 1000
2902 000088FE 7208                <1>     jb      short pass_dfc
2903 00008900 47                  <1>     inc     edi
2904 00008901 663D1027            <1>     cmp     ax, 10000
2905 00008905 7201                <1>     jb      short pass_dfc
2906 00008907 47                  <1>     inc     edi
2907                              <1> pass_dfc:
2908                              <1>     ;mov    cx, 10
2909 00008908 886F01              <1>     mov     [edi+1], ch ; 00
2910                              <1> loc_dfc_rediv:
2911                              <1>     ;;xor   dx, dx
2912                              <1>     ;xor    dl, dl
2913                              <1>     ;div    cx
2914                              <1>     ; 27/07/022
2915 0000890B 31D2                <1>     xor     edx, edx
2916 0000890D F7F1                <1>     div     ecx
2917 0000890F 80C230              <1>     add     dl, '0'
2918 00008912 8817                <1>     mov     [edi], dl
2919 00008914 4F                  <1>     dec     edi
2920                              <1>     ;or     ax, ax
2921                              <1>     ; 27/07/2022
2922 00008915 09C0                <1>     or      eax, eax
2923 00008917 75F2                <1>     jnz     short loc_dfc_rediv
2924                              <1>
2925 00008919 BF[80800100]        <1>     mov     edi, TFS_Dec_End
2926                              <1>         ;mov    byte [edi], 0
2927 0000891E A1[6E800100]        <1>     mov     eax, [Total_FSize]
2928                              <1>     ;mov    ecx, 10
2929                              <1> rediv_tfs_hex:
2930 00008923 29D2                <1>     sub     edx, edx ; 27/07/2022
2931                              <1>     ;sub    dl, dl
2932 00008925 F7F1                <1>     div     ecx
2933 00008927 80C230              <1>     add     dl, '0'
2934 0000892A 4F                  <1>     dec     edi
2935 0000892B 8817                <1>     mov     [edi], dl
2936 0000892D 21C0                <1>     and     eax, eax
2937 0000892F 75F2                <1>     jnz     short rediv_tfs_hex
2938                              <1>
2939 00008931 893D[72800100]      <1>     mov     [TFS_Dec_Begin], edi
2940 00008937 BE[3C320100]        <1>     mov     esi, Decimal_File_Count_Header
2941 0000893C E869E4FFFF          <1>     call    print_msg
2942 00008941 BE[44320100]        <1>     mov     esi, str_files
2943 00008946 E85FE4FFFF          <1>     call    print_msg
2944 0000894B BE[55320100]        <1>     mov     esi, str_dirs
2945 00008950 E855E4FFFF          <1>     call    print_msg
2946 00008955 8B35[72800100]      <1>     mov     esi, [TFS_Dec_Begin]
2947 0000895B E84AE4FFFF          <1>     call    print_msg
2948 00008960 BE[66320100]        <1>     mov     esi, str_bytes
2949                              <1>     ;call   print_msg
2950                              <1>     ;retn
2951                              <1>     ; 27/07/2022
2952 00008965 E940E4FFFF          <1>     jmp     print_msg
2953                              <1>
2954                              <1> find_first_file:
2955                              <1>     ; 28/07/2022 (TRDOS 386 Kernel v2.0.5)
2956                              <1>     ; 11/02/2016
```

```
                                <1>     ; 10/02/2016
                                <1>     ; 08/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>     ; 09/10/2011
                                <1>     ; 17/09/2009
                                <1>     ; 2005
                                <1>     ; INPUT ->
                                <1>     ;       ESI = ASCIIZ File/Dir Name Address (in Current Directory)
                                <1>     ;       AL = Attributes AND mask (The AND result must be equal to AL)
                                <1>     ;             bit 0 = Read Only
                                <1>     ;             bir 1 = Hidden
                                <1>     ;             bit 2 = System
                                <1>     ;             bit 3 = Volume Label
                                <1>     ;             bit 4 = Directory
                                <1>     ;             bit 5 = Archive
                                <1>     ;             bit 6 = Reserved, must be 0
                                <1>     ;             bit 7 = Reserved, must be 0
                                <1>     ;        AH = Attributes Negative AND mask (The AND result must be ZERO)
                                <1>     ;
                                <1>     ; OUTPUT ->
                                <1>     ;       CF = 1 -> Error, Error Code in EAX (AL)
                                <1>     ;       CF = 0 ->
                                <1>     ;            ESI = Directory Entry (FindFile_DirEntry) Location
                                <1>     ;            EDI = Directory Buffer Directory Entry Location
                                <1>     ;            EAX = File Size
                                <1>     ;             BL = Attributes of The File/Directory
                                <1>     ;             BH = Long Name Yes/No Status (>0 is YES)
                                <1>     ;              DX > 0 : Ambiguous filename chars are used
                                <1>     ;
                                <1>     ; (EAX, EBX, ECX, EDX, ESI, EDI will be changed)
                                <1>
0000896A 66A3[32800100]         <1>     mov     [FindFile_AttributesMask], ax
00008970 BF[34800100]           <1>     mov     edi, FindFile_DirEntry ; TR-DOS Fullfilename formatted buffer
00008975 31C0                   <1>     xor     eax, eax
                                <1>     ;mov    ecx, 11
                                <1>     ; 28/07/2022
00008977 31C9                   <1>     xor     ecx, ecx
00008979 B10B                   <1>     mov     cl, 11
0000897B F3AB                   <1>     rep     stosd   ; 44 bytes
                                <1>     ;stosw          ; +2 bytes
                                <1>
0000897D BF[24800100]           <1>     mov     edi, FindFile_Name ; FFF structure, offset 66
00008982 39FE                   <1>     cmp     esi, edi
00008984 7408                   <1>     je      short loc_fff_mfn_ok
00008986 89FA                   <1>     mov     edx, edi
                                <1>     ; move 13 bytes
00008988 A5                     <1>     movsd
00008989 A5                     <1>     movsd
0000898A A5                     <1>     movsd
0000898B AA                     <1>     stosb
0000898C 89D6                   <1>     mov     esi, edx
                                <1> loc_fff_mfn_ok:
0000898E BF[D37F0100]           <1>     mov     edi, Dir_Entry_Name ; Dir Entry Format File Name
00008993 E80C1F0000             <1>     call    convert_file_name
00008998 89FE                   <1>     mov     esi, edi ; offset Dir_Entry_Name
                                <1>
0000899A 66A1[32800100]         <1>     mov     ax, [FindFile_AttributesMask]
                                <1>     ;xor    ecx, ecx
000089A0 30C9                   <1>     xor     cl, cl
000089A2 E8511C0000             <1>     call    locate_current_dir_file
000089A7 726D                   <1>     jc      short loc_fff_retn
                                <1>     ; EDI = Directory Entry
                                <1>     ; EBX = Directory Buffer Entry Index/Number
                                <1>
                                <1> loc_fff_fnf_ln_check:
000089A9 30ED                   <1>     xor     ch, ch
000089AB 80F60F                 <1>     xor     dh, 0Fh
000089AE 7408                   <1>     jz      short loc_fff_longname_yes
000089B0 882D[31800100]         <1>     mov     [FindFile_LongNameYes], ch ; 0
000089B6 EB0C                   <1>     jmp     short loc_fff_longname_no
                                <1>
                                <1> loc_fff_longname_yes:
                                <1>     ;inc    byte [FindFile_LongNameYes]
000089B8 8A0D[3E7F0100]         <1>     mov     cl, [LFN_EntryLength]
000089BE 880D[31800100]         <1>     mov     [FindFile_LongNameEntryLength], cl ; FindFile_LongNameYes
                                <1>
                                <1> loc_fff_longname_no:
                                <1>     ;mov    bx, [DirBuff_CurrentEntry]
000089C4 66891D[5C800100]       <1>     mov     [FindFile_DirEntryNumber], bx
                                <1>     ;mov    dx, ax ; Ambiguous Filename chars used sign > 0
                                <1>     ; 28/07/2022
000089CB 89C2                   <1>     mov     edx, eax
                                <1>
000089CD A0[42770100]           <1>     mov     al, [Current_Drv]
000089D2 A2[E27F0100]           <1>     mov     [FindFile_Drv], al
                                <1>
000089D7 A1[3C770100]           <1>     mov     eax, [Current_Dir_FCluster]
000089DC A3[54800100]           <1>     mov     [FindFile_DirFirstCluster], eax
                                <1>
000089E1 A1[6E7E0100]           <1>     mov     eax, [DirBuff_Cluster]
000089E6 A3[58800100]           <1>     mov     [FindFile_DirCluster], eax
                                <1>
000089EB 66FF05[5E800100]       <1>     inc     word [FindFile_MatchCounter]
                                <1>
000089F2 89FB                   <1>     mov     ebx, edi
000089F4 89FE                   <1>     mov     esi, edi
000089F6 BF[34800100]           <1>     mov     edi, FindFile_DirEntry
000089FB 89F8                   <1>     mov     eax, edi
000089FD B108                   <1>     mov     cl, 8
000089FF F3A5                   <1>     rep     movsd
00008A01 89C6                   <1>     mov     esi, eax
00008A03 89DF                   <1>     mov     edi, ebx
                                <1>
00008A05 A1[50800100]           <1>     mov     eax, [FindFile_DirEntry+28] ; File Size
                                <1>
00008A0A 8A1D[3F800100]         <1>     mov     bl, [FindFile_DirEntry+11] ; File Attributes
00008A10 8A3D[31800100]         <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:
00008A16 C3                     <1>     retn
                                <1>
                                <1> find_next_file:
                                <1>     ; 19/12/2025 (TRDOS 386 Kernel v2.0.10)
                                <1>     ; 28/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <1>     ; 15/10/2016
                                <1>     ; 10/02/2016
                                <1>     ; 08/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>     ; 06/02/2011
                                <1>     ; 17/09/2009
                                <1>     ; 2005
```

```
                                <1>     ; INPUT ->
                                <1>     ;       NONE, Find First File Parameters
                                <1>     ; OUTPUT ->
                                <1>     ;       CF = 1 -> Error, Error Code in EAX (AL)
                                <1>     ;       CF = 0 ->
                                <1>     ;           ESI = Directory Entry (FindFile_DirEntry) Location
                                <1>     ;           EDI = Directory Buffer Directory Entry Location
                                <1>     ;           EAX = File Size
                                <1>     ;             BL = Attributes of The File/Directory
                                <1>     ;             BH = Long Name Yes/No Status (>0 is YES)
                                <1>     ;              DX > 0 : Ambiguous filename chars are used
                                <1>     ;
                                <1>     ; (EAX, EBX, ECX, EDX, ESI, EDI will be changed)
                                <1>
00008A17 66833D[5E800100]00     <1>     cmp     word [FindFile_MatchCounter], 0
00008A1F 7707                   <1>     ja      short loc_start_search_next_file
                                <1>
                                <1> loc_fnf_stc_retn:
00008A21 F9                     <1>     stc
                                <1> loc_fnf_ax12h_retn:
00008A22 B80C000000             <1>     mov     eax, 12 ; No More files
                                <1> ;loc_fnf_retn:
00008A27 C3                     <1>     retn
                                <1>
                                <1> loc_start_search_next_file:
00008A28 668B1D[5C800100]       <1>     mov     bx, [FindFile_DirEntryNumber]
                                <1>     ;inc    bx
                                <1>     ; 28/07/2022
00008A2F 43                     <1>     inc     ebx
00008A30 663B1D[6C7E0100]       <1>     cmp     bx, [DirBuff_LastEntry]
00008A37 7719                   <1>     ja      short loc_cont_search_next_file
                                <1>
                                <1> loc_fnf_search:
00008A39 BE[D37F0100]           <1>     mov     esi, Dir_Entry_Name
00008A3E 66A1[32800100]         <1>     mov     ax, [FindFile_AttributesMask]
                                <1>     ;xor    cx, cx
                                <1>     ; 28/07/2022
00008A44 31C9                   <1>     xor     ecx, ecx
00008A46 E8881C0000             <1>     call    find_directory_entry
                                <1>     ;jnc    loc_fff_fnf_ln_check
                                <1>     ; 28/07/2022
00008A4B 7205                   <1>     jc      short loc_cont_search_next_file
00008A4D E957FFFFFF             <1>     jmp     loc_fff_fnf_ln_check
                                <1>
                                <1> loc_cont_search_next_file:
00008A52 31DB                   <1>     xor     ebx, ebx
00008A54 8A3D[42770100]         <1>     mov     bh, [Current_Drv]
00008A5A BE00010900             <1>     mov     esi, Logical_DOSDisks
00008A5F 01DE                   <1>     add     esi, ebx
                                <1>
00008A61 803D[40770100]00       <1>     cmp     byte [Current_Dir_Level], 0
                                <1> ; 19/12/2025
                                <1> %if 0
                                <1>     jna     short loc_fnf_check_FAT_type
                                <1>
                                <1>     cmp     byte [esi+LD_FATType], 1
                                <1>     jb      short loc_fnf_ax12h_retn
                                <1>
                                <1>     jmp     short loc_fnf_check_next_cluster
                                <1> %else
                                <1>     ; 19/12/2025
00008A68 7706                   <1>     ja      short loc_fnf_check_next_cluster
                                <1> %endif
                                <1>
                                <1> loc_fnf_check_FAT_type:
00008A6A 807E0303               <1>     cmp     byte [esi+LD_FATType], 3
00008A6E 72B2                   <1>     jb      short loc_fnf_ax12h_retn
                                <1>
                                <1> loc_fnf_check_next_cluster:
00008A70 A1[6E7E0100]           <1>     mov     eax, [DirBuff_Cluster]
00008A75 E8FF340000             <1>     call    get_next_cluster
00008A7A 7306                   <1>     jnc     short loc_fnf_load_next_dir_cluster
00008A7C 09C0                   <1>     or      eax, eax
00008A7E 74A1                   <1>     jz      short loc_fnf_stc_retn
                                <1>     ;mov    eax, 17 ;Drive not ready or read error
00008A80 F5                     <1>     cmc     ;stc
                                <1> loc_fnf_retn:
00008A81 C3                     <1>     retn
                                <1>
                                <1> loc_fnf_load_next_dir_cluster:
00008A82 E8AE360000             <1>     call    load_FAT_sub_directory
00008A87 72F8                   <1>     jc      short loc_fnf_retn
                                <1>     ;xor    bx, bx
                                <1>     ; 28/07/2022
00008A89 31DB                   <1>     xor     ebx, ebx
00008A8B 66891D[5C800100]       <1>     mov     [FindFile_DirEntryNumber], bx
00008A92 EBA5                   <1>     jmp     short loc_fnf_search
                                <1>
                                <1> get_and_print_longname:
                                <1>     ; 19/12/2025 (TRDOS 386 Kernel v2.0.10)
                                <1>     ; 28/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <1>     ; 16/10/2016
                                <1>     ; 13/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>     ; 24/01/2010
                                <1>     ; 17/10/2009 (CMD_INTR.ASM, 'cmp_cmd_longname')
                                <1> get_longname_fchar:
00008A94 803E20                 <1>     cmp     byte [esi], 20h
00008A97 7701                   <1>     ja      short loc_find_longname
                                <1>     ;jb     short loc_longname_retn
                                <1>     ;inc    esi
                                <1>     ;je     short get_longname_fchar
                                <1> ;loc_longname_retn:
00008A99 C3                     <1>     retn
                                <1> loc_find_longname:
00008A9A E86C1F0000             <1>     call    find_longname
00008A9F 731D                   <1>     jnc     short loc_print_longname
                                <1>
00008AA1 08C0                   <1>     or      al, al
00008AA3 7412                   <1>     jz      short loc_longname_not_found
                                <1>
                                <1>     ; 16/10/2016 (15h -> 15, 17)
00008AA5 3C0F                   <1>     cmp     al, 15
                                <1>     ;je     cd_drive_not_ready ; drive not ready
                                <1>     ; 28/07/2022
00008AA7 7505                   <1>     jne     short loc_fln_err2
                                <1> loc_fln_err1:
00008AA9 E92CF8FFFF             <1>     jmp     cd_drive_not_ready
                                <1> loc_fln_err2:
                                <1>                                     ; or
00008AAE 3C11                   <1>     cmp     al, 17                  ; read error
                                <1>     ;je     cd_drive_not_ready
                                <1>     ; 28/07/2022
                                <1>     ;je     short loc_fln_err1
                                <1>
```

```
                        <1> loc_ln_file_dir_not_found:
00008AB0 BE[91310100]    <1>     mov     esi, Msg_File_Directory_Not_Found
                         <1>     ;;call  print_msg
                         <1>         ;;retn
                         <1>     ;jmp    print_msg
                         <1>     ; 28/07/2022
00008AB5 EB1F            <1>     jmp     short loc_lfn_err3
                         <1>
                         <1> loc_longname_not_found:
00008AB7 BE[B0310100]    <1>         mov     esi, Msg_LongName_Not_Found
                         <1>     ;;call  print_msg
                         <1>         ;;retn
                         <1>     ;jmp    print_msg
                         <1>     ; 28/07/2022
00008ABC EB18            <1>     jmp     short loc_lfn_err3
                         <1>
                         <1> loc_print_longname:
                         <1>     ;mov    esi, LongFileName
00008ABE BF[42780100]    <1>     mov     edi, TextBuffer
00008AC3 57              <1>     push    edi
                         <1>
                         <1> ; 19/12/2025
                         <1> %if 0
                         <1>     cmp     al, 0
                         <1>     ja      short loc_print_longname_1
                         <1> loc_print_FS_longname: ; Singlix FS (64 byte ASCIIZ file name)
                         <1>     lodsb
                         <1>     stosb
                         <1>     or      al, al
                         <1>     jnz     short loc_print_FS_longname
                         <1>     jmp     short loc_print_longname_2
                         <1> %endif
                         <1>     ;
                         <1> loc_print_longname_1: ; MS Windows long name (UNICODE chars)
00008AC4 66AD            <1>     lodsw
00008AC6 AA              <1>     stosb
00008AC7 08C0            <1>     or      al, al
00008AC9 75F9            <1>     jnz     short loc_print_longname_1
                         <1>     ;
                         <1> loc_print_longname_2:
00008ACB 5E              <1>     pop     esi
00008ACC E8D9E2FFFF      <1>     call    print_msg
00008AD1 BE[E7380100]    <1>     mov     esi, nextline
                         <1> loc_lfn_err3:
                         <1>     ;call   print_msg
                         <1>     ;retn
00008AD6 E9CFE2FFFF      <1>     jmp     print_msg
                         <1>
                         <1> show_file:
                         <1>     ; 19/12/2025 (TRDOS 386 Kernel v2.0.10)
                         <1>     ; 07/08/2022
                         <1>     ; 28/07/2022 (TRDOS 386 Kernel v2.0.5)
                         <1>     ; 18/02/2016
                         <1>     ; 17/02/2016
                         <1>     ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
                         <1>     ; 13/09/2011 (CMD_INTR.ASM, 'cmp_cmd_show')
                         <1>     ; 08/11/2009
                         <1>
                         <1> loc_show_parse_path_name:
00008ADB BF[E27F0100]    <1>     mov     edi, FindFile_Drv
00008AE0 E8891E0000      <1>     call    parse_path_name
                         <1>     ;jc     loc_cmd_failed
                         <1>     ; 28/07/2022
00008AE5 7305            <1>     jnc     short loc_show_check_filename_exists
                         <1> show_file_err1:
00008AE7 E958F9FFFF      <1>     jmp     loc_cmd_failed
                         <1>
                         <1> loc_show_check_filename_exists:
00008AEC BE[24800100]    <1>     mov     esi, FindFile_Name
00008AF1 803E20          <1>     cmp     byte [esi], 20h
                         <1>     ;jna    loc_cmd_failed
                         <1>     ; 28/07/2022
00008AF4 76F1            <1>     jna     short show_file_err1
                         <1>
                         <1>     ; 15/02/2016 (invalid file name check)
00008AF6 E8CE010000      <1>     call    check_filename
00008AFB 730A            <1>     jnc     short loc_show_change_drv
                         <1>
00008AFD BE[7C320100]    <1>     mov     esi, Msg_invalid_name_chars
00008B02 E9A3E2FFFF      <1>     jmp     print_msg
                         <1>
                         <1> loc_show_change_drv:
00008B07 8A35[42770100]  <1>     mov     dh, [Current_Drv]
00008B0D 8835[9F7E0100]  <1>     mov     [RUN_CDRV], dh
00008B13 8A15[E27F0100]  <1>     mov     dl, [FindFile_Drv]
00008B19 38F2            <1>     cmp     dl, dh
00008B1B 740C            <1>     je      short loc_show_change_directory
00008B1D E8BEEBFFFF      <1>     call    change_current_drive
                         <1>     ;;jc    loc_file_rw_cmd_failed
                         <1>     ;jc     loc_run_cmd_failed
                         <1>     ; 28/07/2022
00008B22 7305            <1>     jnc     short loc_show_change_directory
                         <1> show_file_err2:
00008B24 E946F9FFFF      <1>     jmp     loc_run_cmd_failed
                         <1>
                         <1> loc_show_change_directory:
00008B29 803D[E37F0100]20 <1>    cmp     byte [FindFile_Directory], 20h
00008B30 7614            <1>     jna     short loc_findload_showfile
                         <1>
00008B32 FE05[5D2E0100]  <1>     inc     byte [Restore_CDIR]
00008B38 BE[E37F0100]    <1>     mov     esi, FindFile_Directory
00008B3D 30E4            <1>     xor     ah, ah ; CD_COMMAND sign -> 0
00008B3F E897180000      <1>     call    change_current_directory
                         <1>     ;;jc    loc_file_rw_cmd_failed
                         <1>     ;jc     loc_run_cmd_failed
                         <1>     ; 28/07/2022
00008B44 72DE            <1>     jc      short show_file_err2
                         <1>
                         <1> ;loc_show_change_prompt_dir_string:
                         <1>     ;call   change_prompt_dir_string
                         <1>
                         <1> loc_findload_showfile:
                         <1>     ; 15/02/2016
00008B46 BE[24800100]    <1>     mov     esi, FindFile_Name
00008B4B BF[D37F0100]    <1>     mov     edi, Dir_Entry_Name ; Dir Entry Format File Name
00008B50 E84F1D0000      <1>     call    convert_file_name
00008B55 89FE            <1>     mov     esi, edi ; offset Dir_Entry_Name
                         <1>
00008B57 28C0            <1>     sub     al, al  ; Attrib AND mask = 0
                         <1>     ; Directory attribute : 10h
                         <1>     ; Volume name attribute: 8h
00008B59 B418            <1>     mov     ah, 00011000b ; 18h (Attrib NAND, AND --> zero mask)
                         <1>     ;
                         <1>     ;xor    cx, cx
```

```
                                <1>     ; 28/07/2022
00008B5B 31C9                   <1>     xor     ecx, ecx
00008B5D E8961A0000             <1>     call    locate_current_dir_file
                                <1>     ;;jc    loc_file_rw_cmd_failed
                                <1>     ;jc     loc_run_cmd_failed
                                <1>     ; 28/07/2022
00008B62 72C0                   <1>     jc      short show_file_err2
                                <1>
                                <1> loc_show_load_file:
                                <1>     ; EDI = Directory Entry
00008B64 668B4714               <1>     mov     ax, [edi+DirEntry_FstClusHI] ; First Cluster High Word
00008B68 C1E010                 <1>     shl     eax, 16
00008B6B 668B471A               <1>     mov     ax, [edi+DirEntry_FstClusLO] ; First Cluster Low Word
00008B6F A3[8C800100]           <1>     mov     [Show_Cluster], eax
00008B74 8B471C                 <1>     mov     eax, [edi+DirEntry_FileSize] ; File Size
00008B77 21C0                   <1>     and     eax, eax ; Empty file !
                                <1>     ;jz     end_of_show_file
                                <1>     ; 28/07/2022
00008B79 7505                   <1>     jnz     short loc_show_load_file_set_size
00008B7B E994000000             <1>     jmp     end_of_show_file
                                <1>
                                <1> loc_show_load_file_set_size: ; 28/07/2022
00008B80 A3[90800100]           <1>     mov     [Show_FileSize], eax
00008B85 31C0                   <1>     xor     eax, eax
00008B87 A3[94800100]           <1>     mov     [Show_FilePointer], eax ; 0
00008B8C 66A3[98800100]         <1>     mov     [Show_ClusterPointer], ax ; 0
00008B92 29DB                   <1>     sub     ebx, ebx
00008B94 8A3D[42770100]         <1>     mov     bh, [Current_Drv]
00008B9A BE00010900             <1>     mov     esi, Logical_DOSDisks
00008B9F 01DE                   <1>     add     esi, ebx
00008BA1 8935[88800100]         <1>     mov     [Show_LDDDT], esi ; Logical DOS Drv Description Table addr
                                <1>
                                <1> ; 19/12/2025
                                <1> %if 0
                                <1>     cmp     byte [esi+LD_FATType], 0
                                <1>     ja      short loc_show_calculate_cluster_size
                                <1>     ; Singlix FS
                                <1>     ; First Cluster Number is FDT number (in compatibility buffer)
                                <1>     mov     edx, [Show_Cluster] ; Compatibility dir. buffer value (FDT)
                                <1>     mov     [Show_FDT], edx
                                <1>     ; 19/12/2025
                                <1>     ;xor    eax, eax
                                <1>     mov     [Show_Cluster], eax ; Sector index  = 0
                                <1>                                 ; (next time it will be 1)
                                <1> %endif
                                <1>
                                <1> loc_show_calculate_cluster_size:
                                <1>     ;mov    bx, [esi+LD_BPB+BPB_BytsPerSec] ; FAT 12-16-32 (512)
                                <1>     ; 19/12/2025
00008BA7 66BB0002               <1>     mov     bx, 512
                                <1>     ; BX = 512 = [esi+LD_FS_BytesPerSec] ; Singlix FS
00008BAB 8A4613                 <1>     mov     al, [esi+LD_BPB+BPB_SecPerClust] ; FAT 12-16-32 (<= 128)
                                <1>     ; AL = 1 = [esi+LD_FS_Reserved2] ; SectPerClust for Singlix FS
00008BAE F7E3                   <1>     mul     ebx
                                <1>
                                <1>     ;cmp    eax, 65536 ; non-compatible (very big) cluster size
                                <1>     ;ja     short end_of_show_file
00008BB0 66A3[9A800100]         <1>     mov     [Show_ClusterSize], ax
                                <1>
                                <1> loc_start_show_file:
00008BB6 BE[E7380100]           <1>     mov     esi, nextline
00008BBB E8EAE1FFFF             <1>     call    print_msg
                                <1>
00008BC0 A1[8C800100]           <1>     mov     eax, [Show_Cluster]
00008BC5 C605[9C800100]17       <1>     mov     byte [Show_RowCount], 23
                                <1>
                                <1>     ; 17/02/2016
00008BCC 8B35[88800100]         <1>     mov     esi, [Show_LDDDT]
                                <1>
                                <1>     ; 07/08/2022
                                <1> loc_show_next_cluster:
                                <1>     ; 15/02/2016
00008BD2 BB00000700             <1>     mov     ebx, Cluster_Buffer ; 70000h (for current TRDOS 386 version)
                                <1>     ; ESI = Logical DOS drv description table address
00008BD7 E88F350000             <1>     call    read_cluster
                                <1>     ;;jc    loc_file_rw_cmd_failed
                                <1>     ;jc     loc_run_cmd_failed
                                <1>     ; 07/08/2022
00008BDC 7305                   <1>     jnc     short loc_show_nc_rc_ok
00008BDE E98CF8FFFF             <1>     jmp     loc_run_cmd_failed
                                <1> loc_show_nc_rc_ok:
00008BE3 31DB                   <1>     xor     ebx, ebx
                                <1> loc_show_next_byte:
00008BE5 803D[9C800100]00       <1>     cmp     byte [Show_RowCount], 0
00008BEC 7512                   <1>     jne     short pass_show_wait_for_key
00008BEE 30E4                   <1>     xor     ah, ah
00008BF0 E82783FFFF             <1>     call    int16h
00008BF5 3C1B                   <1>     cmp     al, 1Bh
                                <1>     ;jne    short pass_exit_show
                                <1>     ; 28/07/2022
00008BF7 741B                   <1>     je      short end_of_show_file
                                <1>
                                <1> ;end_of_show_file:
                                <1> ;pass_show_file:
                                <1> ;   mov     esi, nextline
                                <1> ;   call    print_msg
                                <1> ;   jmp     loc_file_rw_restore_retn
                                <1>
                                <1> pass_exit_show:
00008BF9 C605[9C800100]14       <1>     mov     byte [Show_RowCount], 20
                                <1> pass_show_wait_for_key:
00008C00 81C300000700           <1>     add     ebx, Cluster_Buffer
00008C06 8A03                   <1>     mov     al, [ebx]
00008C08 3C0D                   <1>     cmp     al, 0Dh
                                <1>     ;jne    loc_show_check_tab_space
                                <1>     ; 28/07/2022
00008C0A 7417                   <1>     je      short loc_show_dec_row_count
00008C0C EB7E                   <1>     jmp     loc_show_check_tab_space
                                <1>
                                <1>     ; 07/08/2022
                                <1> loc_show_next:
                                <1>     ;and    bx, bx ; 65536 -> 0
                                <1>     ; 28/07/2022
00008C0E 21DB                   <1>     and     ebx, ebx
00008C10 75D3                   <1>     jnz     short loc_show_next_byte
00008C12 EBBE                   <1>     jmp      short loc_show_next_cluster
                                <1>
                                <1>     ; 19/12/2025
                                <1>     ; 28/07/2022
                                <1> end_of_show_file:
                                <1> pass_show_file:
00008C14 BE[E7380100]           <1>     mov     esi, nextline
00008C19 E88CE1FFFF             <1>     call    print_msg
00008C1E E915010000             <1>     jmp     loc_file_rw_restore_retn
```

```
3453                                    <1>
3454                                    <1> loc_show_dec_row_count:    ; 28/07/2022
3455 00008C23 FE0D[9C800100]           <1>     dec     byte [Show_RowCount]
3456                                    <1> pass_show_dec_rowcount:
3457 00008C29 B307                     <1>     mov     bl, 7 ; (light gray character color, black background)
3458 00008C2B 8A3D[AE760100]           <1>     mov     bh, [ACTIVE_PAGE] ; [ptty]
3459 00008C31 E8A596FFFF               <1>     call    _write_tty
3460                                    <1> loc_show_check_eof:
3461 00008C36 FF05[94800100]           <1>     inc     dword [Show_FilePointer]
3462 00008C3C A1[94800100]             <1>     mov     eax, [Show_FilePointer]
3463 00008C41 3B05[90800100]           <1>     cmp     eax, [Show_FileSize]
3464 00008C47 73CB                     <1>     jnb     short end_of_show_file
3465 00008C49 66FF05[98800100]         <1>     inc     word [Show_ClusterPointer]
3466 00008C50 0FB71D[98800100]         <1>     movzx   ebx, word [Show_ClusterPointer]
3467                                    <1>
3468                                    <1>     ; 17/02/2016
3469                                    <1>     ; (sector boundary -9 bits- check, 512 = 0)
3470 00008C57 66F7C3FF01               <1>       test    bx, 1FFh ; 1 to 511
3471 00008C5C 7587                     <1>     jnz     short loc_show_next_byte
3472                                    <1>
3473                                    <1>     ; 16/02/2016
3474 00008C5E 8B35[88800100]           <1>     mov     esi, [Show_LDDDT]
3475                                    <1>
3476                                    <1> ; 19/12/2025
3477                                    <1> %if 0
3478                                    <1>     cmp     byte [esi+LD_FATType], 0
3479                                    <1>     ja      short loc_show_check_fat_cluster_size
3480                                    <1>
3481                                    <1>     ; Singlix FS
3482                                    <1>     ; 1 sector, more... (cluster size = 1 sector)
3483                                    <1>     mov     eax, [Show_Cluster]
3484                                    <1>     inc     eax
3485                                    <1>     mov     [Show_Cluster], eax
3486                                    <1>
3487                                    <1>     ; 07/08/2022
3488                                    <1>     jmp     short loc_show_next
3489                                    <1>
3490                                    <1>     ; 28/07/2022
3491                                    <1> end_of_show_file:
3492                                    <1> pass_show_file:
3493                                    <1>     mov     esi, nextline
3494                                    <1>     call    print_msg
3495                                    <1>     jmp     loc_file_rw_restore_retn
3496                                    <1> %endif
3497                                    <1>
3498                                    <1> loc_show_check_fat_cluster_size:
3499                                    <1>     ; 17/02/2016
3500 00008C64 663B1D[9A800100]         <1>     cmp     bx, [Show_ClusterSize] ; cluster size in bytes
3501                                    <1>         ;jb short loc_show_next_byte ; 28/07/2022
3502                                    <1>     ; 07/08/2022
3503 00008C6B 7305                     <1>     jnb     short loc_show_file_cluster_ok
3504 00008C6D E973FFFFFF               <1>     jmp     loc_show_next_byte
3505                                    <1>
3506                                    <1> loc_show_file_cluster_ok:
3507 00008C72 66C705[98800100]00-      <1>     mov     word [Show_ClusterPointer], 0
3507 00008C7A 00                       <1>
3508                                    <1>
3509 00008C7B A1[8C800100]             <1>     mov     eax, [Show_Cluster]
3510                                    <1>     ;mov    esi, [Show_LDDDT]
3511                                    <1> loc_show_get_next_cluster:
3512 00008C80 E8F4320000               <1>     call    get_next_cluster
3513                                    <1>     ;;jc    loc_file_rw_cmd_failed
3514                                    <1>     ;jc     loc_run_cmd_failed
3515                                    <1>     ; 28/07/2022
3516 00008C85 7338                     <1>     jnc     short loc_show_update_ccluster
3517 00008C87 E9E3F7FFFF               <1>     jmp     loc_run_cmd_failed
3518                                    <1>
3519                                    <1> loc_show_check_tab_space:
3520 00008C8C 3C09                     <1>     cmp     al, 09h
3521                                    <1>     ;jne    short pass_show_dec_rowcount ; 28/07/2022
3522                                    <1>     ; 07/08/2022
3523 00008C8E 7402                     <1>     je      short loc_show_put_tab_space
3524 00008C90 EB97                     <1>     jmp     pass_show_dec_rowcount
3525                                    <1> loc_show_put_tab_space:
3526 00008C92 8A3D[AE760100]           <1>     mov     bh, [ACTIVE_PAGE] ; [ptty]
3527 00008C98 E8DC92FFFF               <1>     call    get_cpos
3528                                    <1>     ; dl = cursor column
3529 00008C9D 80E207                   <1>     and     dl, 7 ; 18/02/2016
3530                                    <1>     ;shr    bh, 1 ; [ACTIVE_PAGE]
3531 00008CA0 8A3D[AE760100]           <1>     mov     bh, [ACTIVE_PAGE]
3532 00008CA6 B307                     <1>     mov     bl, 7 ; color attribute
3533                                    <1> loc_show_put_space_chars:
3534 00008CA8 B020                     <1>     mov     al, 20h ; space
3535                                    <1>     ;mov    bh, [ACTIVE_PAGE] ; [ptty]
3536                                    <1>     ;mov    bl, 7 ; color attribute
3537                                    <1>     ;push   dx
3538 00008CAA 52                       <1>     push    edx ; 29/12/2017
3539 00008CAB E82B96FFFF               <1>     call    _write_tty
3540 00008CB0 5A                       <1>     pop     edx ; 29/12/2017
3541                                    <1>     ;pop    dx
3542                                    <1>     ; 18/02/2016
3543 00008CB1 80FA07                   <1>     cmp     dl, 7
3544                                    <1>     ;jnb    short loc_show_check_eof ; 28/07/2022
3545                                    <1>     ; 07/08/2022
3546 00008CB4 7205                     <1>     jb      short loc_show_next_tab_space
3547 00008CB6 E97BFFFFFF               <1>     jmp     loc_show_check_eof
3548                                    <1> loc_show_next_tab_space:
3549 00008CBB FEC2                     <1>     inc     dl
3550 00008CBD EBE9                     <1>     jmp     short loc_show_put_space_chars
3551                                    <1>
3552                                    <1> loc_show_update_ccluster:
3553 00008CBF A3[8C800100]             <1>     mov     [Show_Cluster], eax
3554 00008CC4 E909FFFFFF               <1>         jmp     loc_show_next_cluster
3555                                    <1>
3556                                    <1> check_filename:
3557                                    <1>     ; 28/07/2022 (TRDOS 386 Kernel v2.0.5)
3558                                    <1>     ; 10/10/2016
3559                                    <1>     ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
3560                                    <1>     ; 07/08/2010 (FILE.ASM, 'proc_check_filename')
3561                                    <1>     ; 10/07/2010
3562                                    <1>     ; Derived from 'proc_check_filename'
3563                                    <1>     ; in the old TRDOS.ASM (09/02/2005).
3564                                    <1>     ;
3565                                    <1>     ; INPUT ->
3566                                    <1>     ;       ESI = Dot File Name Location
3567                                    <1>     ; OUTPUT ->
3568                                    <1>     ;       cf = 1 -> error code in AL
3569                                    <1>     ;            AL = ERR_INV_FILE_NAME (=26)
3570                                    <1>     ;                 Invalid file name chars
3571                                    <1>     ;       cf = 0 -> valid file name
3572                                    <1>     ;
3573                                    <1>     ; (EAX, ECX, EDI will be changed)
3574                                    <1>
3575                                    <1> check_invalid_filename_chars:
```

```
                                <1>     ; 28/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <1>     ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>     ; 10/07/2010 (FILE.ASM, 'proc_check_invalid_filename_chars')
                                <1>     ; 10/02/2010
                                <1>     ; Derived from 'proc_check_invalid_filename_chars'
                                <1>     ; in the old TRDOS.ASM (09/02/2005).
                                <1>     ;
                                <1>     ; INPUT ->
                                <1>     ;       ESI = ASCIIZ FileName
                                <1>     ; OUTPUT ->
                                <1>     ;       cf = 1 -> invalid
                                <1>     ;       cf = 0 -> valid
                                <1>     ;
                                <1>     ;(EAX, ECX, EDI will be changed)
                                <1>
00008CC9 56                     <1>     push    esi
                                <1>
00008CCA BF[652F0100]           <1>         mov     edi, invalid_fname_chars
00008CCF AC                     <1>     lodsb
                                <1> check_filename_next_char:
00008CD0 B914000000             <1>     mov     ecx, sizeInvFnChars
00008CD5 BF[652F0100]           <1>     mov     edi, invalid_fname_chars
                                <1> loc_scan_invalid_filename_char:
00008CDA AE                     <1>     scasb
00008CDB 741E                   <1>     je      short loc_invalid_filename_stc
00008CDD E2FB                   <1>     loop    loc_scan_invalid_filename_char
00008CDF AC                     <1>     lodsb
00008CE0 3C1F                   <1>     cmp     al, 1Fh  ; 20h and above
00008CE2 77EC                   <1>     ja      short check_filename_next_char
                                <1>
                                <1> check_filename_dot:
00008CE4 8B3424                 <1>     mov     esi, [esp]
                                <1>
00008CE7 B421                   <1>     mov     ah, 21h
                                <1>     ;mov    ecx, 8
                                <1>     ; 28/07/2022
00008CE9 29C9                   <1>     sub     ecx, ecx
00008CEB B108                   <1>     mov     cl, 8
                                <1> loc_check_filename_next_char:
00008CED AC                     <1>     lodsb
00008CEE 3C2E                   <1>     cmp     al, 2Eh
00008CF0 7511                   <1>     jne     short pass_check_fn_dot_check
                                <1> loc_check_filename_ext_0:
00008CF2 AC                     <1>     lodsb
00008CF3 38E0                   <1>     cmp     al, ah ; 21h
00008CF5 7205                   <1>     jb      short loc_invalid_filename
00008CF7 3C2E                   <1>     cmp     al, 2Eh
00008CF9 7519                   <1>     jne     short loc_check_filename_ext_1
                                <1>
                                <1> loc_invalid_filename_stc:
                                <1> loc_check_fn_stc_rtn:
00008CFB F9                     <1>     stc
                                <1> loc_invalid_filename:
                                <1>     ; 10/10/2016 (0Bh -> 26)
00008CFC B81A000000             <1>     mov     eax, ERR_INV_FILE_NAME ; (=26)
                                <1>     ; Invalid file name chars
                                <1> loc_check_fn_rtn:
00008D01 5E                     <1>     pop     esi
00008D02 C3                     <1>     retn
                                <1>
                                <1> pass_check_fn_dot_check:
00008D03 38E0                   <1>     cmp     al, ah ; 21h
00008D05 7224                   <1>     jb      short loc_check_fn_clc_rtn
00008D07 E2E4                   <1>     loop    loc_check_filename_next_char
00008D09 AC                     <1>     lodsb
00008D0A 38E0                   <1>     cmp     al, ah ; 21h
00008D0C 721D                   <1>     jb      short loc_check_fn_clc_rtn
00008D0E 3C2E                   <1>     cmp     al, 2Eh
00008D10 75E9                   <1>     jne     short loc_check_fn_stc_rtn
00008D12 EBDE                   <1>     jmp     short loc_check_filename_ext_0
                                <1>
                                <1> loc_check_filename_ext_1:
00008D14 AC                     <1>     lodsb
00008D15 38E0                   <1>     cmp     al, ah ; 21h
00008D17 7212                   <1>     jb      short loc_check_fn_clc_rtn
00008D19 3C2E                   <1>     cmp     al, 2Eh
00008D1B 74DE                   <1>     je      short loc_check_fn_stc_rtn
00008D1D AC                     <1>     lodsb
00008D1E 38E0                   <1>     cmp     al, ah ; 21h
00008D20 7209                   <1>     jb      short loc_check_fn_clc_rtn
00008D22 3C2E                   <1>     cmp     al, 2Eh
00008D24 74D5                   <1>     je      short loc_check_fn_stc_rtn
00008D26 AC                     <1>     lodsb
00008D27 38E0                   <1>     cmp     al, ah ; 21h
00008D29 73D0                   <1>     jnb     short loc_check_fn_stc_rtn
                                <1>
                                <1> loc_check_fn_clc_rtn:
00008D2B 5E                     <1>     pop     esi
00008D2C F8                     <1>     clc
00008D2D C3                     <1>     retn
                                <1>
                                <1> loc_print_deleted_message:
00008D2E BE[51330100]           <1>     mov     esi, Msg_Deleted
00008D33 E872E0FFFF             <1>     call    print_msg
                                <1>
                                <1>     ;clc
                                <1>
                                <1> loc_file_rw_restore_retn:
                                <1>     ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>     ; 28/02/2010 (CMD_INTR.ASM)
                                <1> loc_file_rw_cmd_failed:
00008D38 9C                     <1>     pushf
00008D39 E899F7FFFF             <1>     call    restore_cdir_after_cmd_fail
00008D3E 9D                     <1>     popf
00008D3F 720D                   <1>     jc      short loc_file_rw_check_write_fault
00008D41 C3                     <1>     retn
                                <1>
                                <1> loc_permission_denied:
                                <1>     ; 27/02/2016
00008D42 BE[5E330100]           <1>     mov     esi, Msg_Permission_Denied
00008D47 E85EE0FFFF             <1>     call    print_msg
00008D4C EBEA                   <1>     jmp     short loc_file_rw_restore_retn
                                <1>
                                <1> loc_file_rw_check_write_fault:
                                <1>     ;cmp    al, 1Dh ; Write Fault
                                <1>         cmp al, 18 ; 05/11/2016
                                <1>     ;jne    loc_run_cmd_failed_cmp_al
                                <1>     ; 28/07/2022
00008D50 7405                   <1>     je      short loc_file_rw_fault
00008D52 E91DF7FFFF             <1>     jmp     loc_run_cmd_failed_cmp_al
                                <1>
                                <1> loc_file_rw_fault:
00008D57 BE[46310100]           <1>     mov     esi, Msg_Not_Ready_Write_Err
                                <1>     ;call   print_msg
```

```
3700                                    <1>     ;retn
3701 00008D5C E949E0FFFF                <1>     jmp     print_msg
3702                                    <1>
3703                                    <1> make_directory:
3704                                    <1>     ; 28/07/2022 (TRDOS 386 Kernel v2.0.5)
3705                                    <1>     ; 21/02/2016 (TRDOS 386 = TRDOS v2.0)
3706                                    <1>     ; 12/03/2011 (CMD_INTR.ASM, 'cmp_cmd_mkdir')
3707                                    <1>     ; 14/08/2010
3708                                    <1>     ; 10/07/2010
3709                                    <1>     ; 29/11/2009
3710                                    <1>     ;
3711                                    <1> get_mkdir_fchar:
3712                                    <1>     ; esi = directory name
3713 00008D61 803E20                    <1>     cmp     byte [esi], 20h
3714 00008D64 7701                      <1>         ja  short loc_mkdir_parse_path_name
3715                                    <1>
3716                                    <1> loc_mkdir_nodirname_retn:
3717 00008D66 C3                        <1>     retn
3718                                    <1>
3719                                    <1> loc_mkdir_parse_path_name:
3720 00008D67 BF[E27F0100]              <1>     mov     edi, FindFile_Drv
3721 00008D6C E8FD1B0000                <1>         call    parse_path_name
3722                                    <1>     ;jc     loc_cmd_failed
3723                                    <1>     ; 28/07/2022
3724 00008D71 7305                      <1>     jnc     short loc_mkdir_check_dirname_exists
3725                                    <1> loc_mkdir_cmd_failed:
3726 00008D73 E9CCF6FFFF                <1>     jmp     loc_cmd_failed
3727                                    <1>
3728                                    <1> loc_mkdir_check_dirname_exists:
3729 00008D78 BE[24800100]              <1>     mov     esi, FindFile_Name
3730 00008D7D 803E20                    <1>     cmp     byte [esi], 20h
3731                                    <1>     ;jna    loc_cmd_failed
3732                                    <1>     ; 28/07/2022
3733 00008D80 76F1                      <1>     jna     short loc_mkdir_cmd_failed
3734 00008D82 8935[A0800100]            <1>     mov     [DelFile_FNPointer], esi
3735 00008D88 E83CFFFFFF                <1>     call    check_filename
3736 00008D8D 725B                      <1>     jc      short loc_mkdir_invalid_dir_name_chars
3737                                    <1>
3738                                    <1> loc_mkdir_drv:
3739 00008D8F 8A35[42770100]            <1>     mov     dh, [Current_Drv]
3740 00008D95 8835[9F7E0100]            <1>     mov     [RUN_CDRV], dh
3741                                    <1>
3742 00008D9B 8A15[E27F0100]            <1>     mov     dl, [FindFile_Drv]
3743 00008DA1 38F2                      <1>     cmp     dl, dh
3744 00008DA3 7409                      <1>     je      short loc_mkdir_change_directory
3745                                    <1>
3746 00008DA5 E836E9FFFF                <1>     call    change_current_drive
3747                                    <1>     ;jc     loc_file_rw_cmd_failed
3748                                    <1>     ; 28/07/2022
3749 00008DAA 7302                      <1>     jnc     short loc_mkdir_change_directory
3750 00008DAC EB8A                      <1>     jmp     loc_file_rw_cmd_failed
3751                                    <1>
3752                                    <1> loc_mkdir_change_directory:
3753 00008DAE 803D[E37F0100]20          <1>     cmp     byte [FindFile_Directory], 20h
3754 00008DB5 7614                      <1>     jna     short loc_mkdir_find_directory
3755                                    <1>
3756 00008DB7 FE05[5D2E0100]            <1>     inc     byte [Restore_CDIR]
3757 00008DBD BE[E37F0100]              <1>     mov     esi, FindFile_Directory
3758 00008DC2 30E4                      <1>     xor     ah, ah ; CD_COMMAND sign -> 0
3759 00008DC4 E812160000                <1>     call    change_current_directory
3760 00008DC9 7226                      <1>     jc      short loc_mkdir_check_error_code
3761                                    <1>
3762                                    <1> ;loc_mkdir_change_prompt_dir_string:
3763                                    <1>     ;call   change_prompt_dir_string
3764                                    <1>
3765                                    <1> loc_mkdir_find_directory:
3766                                    <1>     ;mov    esi, FindFile_Name
3767 00008DCB 8B35[A0800100]            <1>     mov     esi, [DelFile_FNPointer]
3768                                    <1>     ;xor    eax, eax
3769 00008DD1 6631C0                    <1>     xor     ax, ax ; any name (dir, file, volume)
3770 00008DD4 E891FBFFFF                <1>     call    find_first_file
3771 00008DD9 7216                      <1>     jc      short loc_mkdir_check_error_code
3772                                    <1>
3773                                    <1> loc_mkdir_directory_found:
3774 00008DDB BE[A9320100]              <1>     mov     esi, Msg_Name_Exists
3775                                    <1> loc_mkdir_inv_dname_chrs_msg:
3776 00008DE0 E8C5DFFFFF                <1>     call    print_msg
3777 00008DE5 E94EFFFFFF                <1>     jmp     loc_file_rw_restore_retn
3778                                    <1>
3779                                    <1> loc_mkdir_invalid_dir_name_chars:
3780 00008DEA BE[7C320100]              <1>     mov     esi, Msg_invalid_name_chars
3781                                    <1>     ;call   print_msg
3782                                    <1>         ;jmp        loc_file_rw_restore_retn
3783                                    <1>     ; 28/07/2022
3784 00008DEF EBEF                      <1>     jmp     short loc_mkdir_inv_dname_chrs_msg
3785                                    <1>
3786                                    <1> loc_mkdir_check_error_code:
3787 00008DF1 3C02                      <1>     cmp     al, 2
3788                                    <1>     ;je     short loc_mkdir_directory_not_found
3789 00008DF3 7406                      <1>     je      short loc_mkdir_ask_for_yes_no
3790 00008DF5 F9                        <1>     stc
3791 00008DF6 E93DFFFFFF                <1>         jmp     loc_file_rw_cmd_failed
3792                                    <1>
3793                                    <1> loc_mkdir_directory_not_found:
3794                                    <1> loc_mkdir_ask_for_yes_no:
3795 00008DFB BE[CA320100]              <1>     mov     esi, Msg_DoYouWantMkdir
3796 00008E00 E8A5DFFFFF                <1>     call    print_msg
3797 00008E05 8B35[A0800100]            <1>     mov     esi, [DelFile_FNPointer]
3798 00008E0B E89ADFFFFF                <1>     call    print_msg
3799 00008E10 BE[E9320100]              <1>     mov     esi, Msg_YesNo
3800 00008E15 E890DFFFFF                <1>     call    print_msg
3801                                    <1>
3802 00008E1A C605[F3320100]20          <1>     mov     byte [Y_N_nextline], 20h
3803                                    <1>
3804                                    <1> loc_mkdir_ask_again:
3805 00008E21 30E4                      <1>     xor     ah, ah
3806 00008E23 E8F480FFFF                <1>     call    int16h
3807 00008E28 3C1B                      <1>     cmp     al, 1Bh
3808                                    <1>     ;je     short loc_do_not_make_directory
3809 00008E2A 743B                      <1>     je      short loc_mkdir_y_n_escape
3810 00008E2C 24DF                      <1>     and     al, 0DFh ; y -> Y, n -> N
3811 00008E2E 3C59                      <1>     cmp     al, 'Y' ; 'yes'
3812 00008E30 7404                      <1>     je      short loc_mkdir_yes_make_directory
3813 00008E32 3C4E                      <1>     cmp     al, 'N' ; 'no'
3814 00008E34 75EB                      <1>     jne     short loc_mkdir_ask_again
3815                                    <1>
3816                                    <1> loc_do_not_make_directory:
3817                                    <1> loc_mkdir_yes_make_directory:
3818 00008E36 E830000000                <1>     call    y_n_answer ; 29/12/2017
3819                                    <1>     ;cmp    al, 'Y' ; 'yes'
3820                                    <1>     ;cmc
3821                                    <1>         ;jnc        loc_file_rw_restore_retn
3822 00008E3B 3C4E                      <1>     cmp     al, 'N' ; 'no'
3823                                    <1>     ;je     loc_file_rw_restore_retn
```

```
                                <1>     ; 28/07/2022
00008E3D 7505                   <1>     jne     short loc_mkdir_call_make_sub_dir
00008E3F E9F4FEFFFF             <1>     jmp     loc_file_rw_restore_retn
                                <1>
                                <1> loc_mkdir_call_make_sub_dir:
00008E44 8B35[A0800100]         <1>     mov     esi, [DelFile_FNPointer]
00008E4A B110                   <1>     mov     cl, 10h ; Directory attributes
00008E4C E8001C0000             <1>     call    make_sub_directory
                                <1> loc_rename_file_ok: ; 06/03/2016
                                <1>         ;jc loc_file_rw_cmd_failed
                                <1>     ; 28/07/2022
00008E51 7305                   <1>     jnc     short move_source_file_to_dest_OK
00008E53 E9E0FEFFFF             <1>     jmp     loc_file_rw_cmd_failed
                                <1> move_source_file_to_dest_OK:
00008E58 BE[F7320100]           <1>     mov     esi, Msg_OK
00008E5D E848DFFFFF             <1>     call    print_msg
00008E62 E9D1FEFFFF             <1>     jmp     loc_file_rw_restore_retn
                                <1>
                                <1> loc_mkdir_y_n_escape:
00008E67 B04E                   <1>     mov     al, 'N' ; 'no'
00008E69 EBCB                   <1>     jmp     short loc_do_not_make_directory
                                <1>
                                <1> y_n_answer:
                                <1>     ; 29/12/2017
00008E6B A2[F3320100]           <1>     mov     [Y_N_nextline], al
                                <1>     ;push   ax
00008E70 50                     <1>     push    eax
00008E71 BE[F3320100]           <1>     mov     esi, Y_N_nextline
00008E76 E82FDFFFFF             <1>     call    print_msg
00008E7B 58                     <1>     pop     eax
                                <1>     ;pop    ax
00008E7C C3                     <1>     retn
                                <1>
                                <1> delete_directory:
                                <1>     ; 17/07/2025 (TRDOS 386 Kernel v2.0.10)
                                <1>     ; 28/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <1>     ; 29/12/2017
                                <1>     ; 15/10/2016
                                <1>     ; 01/03/2016, 06/03/2016
                                <1>     ; 27/02/2016, 28/02/2016, 29/02/2016
                                <1>     ; 26/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>     ; 16/10/2010 (CMD_INTR.ASM, 'cmp_cmd_rmdir')
                                <1>     ; 05/06/2010
                                <1>     ;
                                <1> get_fchar:
                                <1>     ; esi = directory name
00008E7D 803E20                 <1>     cmp     byte [esi], 20h
00008E80 7701                   <1>         ja  short loc_rmdir_parse_path_name
                                <1>
                                <1> loc_rmdir_nodirname_retn:
00008E82 C3                     <1>     retn
                                <1>
                                <1> loc_rmdir_parse_path_name:
00008E83 BF[E27F0100]           <1>     mov     edi, FindFile_Drv
00008E88 E8E11A0000             <1>     call    parse_path_name
                                <1>     ;jc     loc_cmd_failed
                                <1>     ; 28/07/2022
00008E8D 7305                   <1>     jnc     short loc_rmdir_check_dirname_exists
                                <1> lc_del_dir_failed:
00008E8F E9B0F5FFFF             <1>     jmp     loc_cmd_failed
                                <1>
                                <1> loc_rmdir_check_dirname_exists:
00008E94 BE[24800100]           <1>     mov     esi, FindFile_Name
00008E99 803E20                 <1>     cmp     byte [esi], 20h
                                <1>     ;jna    loc_cmd_failed
                                <1>     ; 28/07/2022
00008E9C 76F1                   <1>     jna     short lc_del_dir_failed
00008E9E 8935[A0800100]         <1>     mov     [DelFile_FNPointer], esi
                                <1>
                                <1> loc_rmdir_drv:
00008EA4 8A35[42770100]         <1>     mov     dh, [Current_Drv]
00008EAA 8835[9F7E0100]         <1>     mov     [RUN_CDRV], dh
                                <1>
                                <1>     ;;;;
                                <1>     ; 17/07/2025 - TRDOS 386 v2.0.10
00008EB0 8835[E8800100]         <1>     mov     [rmdir_drv], dh
00008EB6 8B1D[3C770100]         <1>     mov     ebx, [Current_Dir_FCluster]
00008EBC 891D[E4800100]         <1>     mov     [rmdir_dir_fcluster], ebx
                                <1>     ;;;;
                                <1>
00008EC2 8A15[E27F0100]         <1>     mov     dl, [FindFile_Drv]
00008EC8 38F2                   <1>     cmp     dl, dh
00008ECA 7407                   <1>     je      short loc_rmdir_change_directory
                                <1>
00008ECC E80FE8FFFF             <1>     call    change_current_drive
                                <1>     ;jc     loc_file_rw_cmd_failed
                                <1>     ; 28/07/2022
                                <1>     ;jnc    short loc_rmdir_change_directory
                                <1>     ;jmp    loc_file_rw_cmd_failed
00008ED1 7233                   <1>     jc      short loc_rmdir_chdrv_failed
                                <1>
                                <1> loc_rmdir_change_directory:
00008ED3 803D[E37F0100]20       <1>     cmp     byte [FindFile_Directory], 20h
00008EDA 7614                   <1>     jna     short loc_rmdir_find_directory
                                <1>
00008EDC FE05[5D2E0100]         <1>     inc     byte [Restore_CDIR]
00008EE2 BE[E37F0100]           <1>     mov     esi, FindFile_Directory
00008EE7 30E4                   <1>     xor     ah, ah ; CD_COMMAND sign -> 0
00008EE9 E8ED140000             <1>     call    change_current_directory
00008EEE 7211                   <1>     jc      short loc_rmdir_check_error_code
                                <1>
                                <1> ;loc_rmdir_change_prompt_dir_string:
                                <1>     ;call   change_prompt_dir_string
                                <1>
                                <1> loc_rmdir_find_directory:
                                <1>     ;mov    esi, FindFile_Name
00008EF0 8B35[A0800100]         <1>     mov     esi, [DelFile_FNPointer]
00008EF6 66B81008               <1>     mov     ax, 0810h ; Only directories
00008EFA E86BFAFFFF             <1>     call    find_first_file
00008EFF 730A                   <1>     jnc     short loc_rmdir_ambgfn_check
                                <1>
                                <1> loc_rmdir_check_error_code:
00008F01 3C02                   <1>     cmp     al, 2
00008F03 740B                   <1>     je      short loc_rmdir_directory_not_found
00008F05 F9                     <1>     stc
                                <1> loc_rmdir_chdrv_failed:
00008F06 E92DFEFFFF             <1>     jmp     loc_file_rw_cmd_failed
                                <1>
                                <1> loc_rmdir_ambgfn_check:
00008F0B 6621D2                 <1>     and     dx, dx ; Ambiguous filename chars used sign (DX>0)
00008F0E 740F                   <1>     jz      short loc_rmdir_directory_found
                                <1>
                                <1> loc_rmdir_directory_not_found:
00008F10 BE[68310100]           <1>     mov     esi, Msg_Dir_Not_Found
```

```
3948 00008F15 E890DEFFFF          <1>     call    print_msg
3949                              <1>
3950 00008F1A E919FEFFFF          <1>     jmp     loc_file_rw_restore_retn
3951                              <1>
3952                              <1> loc_rmdir_directory_found:
3953 00008F1F 80E307              <1>     and     bl, 07h ; Attributes
3954                              <1>     ;jnz    loc_permission_denied
3955                              <1>     ; 28/07/2022
3956 00008F22 7405                <1>     jz      short loc_rmdir_save_lnel
3957 00008F24 E919FEFFFF          <1>     jmp     loc_permission_denied
3958                              <1>
3959                              <1> loc_rmdir_save_lnel: ; 28/02/2016
3960                              <1>         ;mov bh, [LongName_EntryLength]
3961 00008F29 883D[AA800100]      <1>     mov     [DelFile_LNEL], bh ; Long name entry length (if > 0)
3962                              <1>     ; edi = Directory Entry Offset (DirBuff)
3963                              <1>     ; esi = Directory Entry (FFF Structure)
3964                              <1>     ;mov    [DelFile_DirEntryAddr], edi ; not required
3965                              <1>     ;mov    ax, [edi+20] ; First Cluster High Word
3966                              <1>         ;shl        eax, 16
3967                              <1>     ;mov    ax, [edi+26] ; First Cluster Low Word
3968                              <1>     ; ROOT Dir First Cluster = 0
3969                              <1>         ;cmp        eax, 2
3970                              <1>     ;jb     loc_update_direntry_1
3971                              <1>
3972                              <1> pass_rmdir_fc_check:
3973 00008F2F 57                  <1>     push    edi ; * (29/02/2016)
3974                              <1>
3975 00008F30 BE[FD320100]        <1>     mov     esi, Msg_DoYouWantRmDir
3976 00008F35 E870DEFFFF          <1>     call    print_msg
3977 00008F3A 8B35[A0800100]      <1>     mov     esi, [DelFile_FNPointer]
3978 00008F40 E865DEFFFF          <1>     call    print_msg
3979 00008F45 BE[E9320100]        <1>     mov     esi, Msg_YesNo
3980 00008F4A E85BDEFFFF          <1>     call    print_msg
3981                              <1>
3982                              <1> loc_rmdir_ask_again:
3983 00008F4F 30E4                <1>     xor     ah, ah
3984 00008F51 E8C67FFFFF          <1>     call    int16h
3985 00008F56 3C1B                <1>     cmp     al, 1Bh
3986                              <1>     ;je     short loc_do_not_delete_directory
3987 00008F58 742F                <1>         je  short loc_rmdir_y_n_escape ; 06/03/2016
3988 00008F5A 24DF                <1>     and     al, 0DFh
3989 00008F5C A2[F3320100]        <1>     mov     [Y_N_nextline], al
3990 00008F61 3C59                <1>     cmp     al, 'Y'
3991 00008F63 7404                <1>     je      short loc_rmdir_yes_delete_directory
3992 00008F65 3C4E                <1>     cmp     al, 'N'
3993 00008F67 75E6                <1>     jne     short loc_rmdir_ask_again
3994                              <1>
3995                              <1> loc_do_not_delete_directory:
3996                              <1> loc_rmdir_yes_delete_directory:
3997 00008F69 E8FDFEFFFF          <1>     call    y_n_answer ; 29/12/2017
3998 00008F6E 5F                  <1>     pop     edi ; * (29/02/2016)
3999                              <1>     ;cmp    al, 'Y' ; 'yes'
4000                              <1>     ;cmc
4001                              <1>         ;jnc        loc_file_rw_restore_retn
4002 00008F6F 3C4E                <1>     cmp     al, 'N' ; 'no'
4003                              <1>     ;je     loc_file_rw_restore_retn
4004                              <1>     ; 28/07/2022
4005                              <1>     ;jne    short loc_delete_sub_dir
4006                              <1>     ;jmp    loc_file_rw_restore_retn
4007 00008F71 7411                <1>     je      short loc_rmdir_rw_restore_retn
4008                              <1>
4009                              <1> loc_delete_sub_dir:
4010                              <1>     ; 29/12/2017
4011 00008F73 E85E000000          <1>     call    delete_sub_directory
4012 00008F78 7213                <1>     jc      short loc_rmdir_cmd_failed
4013                              <1>
4014                              <1> loc_rmdir_ok:
4015 00008F7A BE[F7320100]        <1>     mov     esi, Msg_OK
4016 00008F7F E826DEFFFF          <1>     call    print_msg
4017                              <1> loc_rmdir_rw_restore_retn: ; 28/07/2022
4018 00008F84 E9AFFDFFFF          <1>     jmp     loc_file_rw_restore_retn
4019                              <1>
4020                              <1> loc_rmdir_y_n_escape:
4021 00008F89 B04E                <1>     mov     al, 'N' ; 'no'
4022 00008F8B EBDC                <1>         jmp     loc_do_not_delete_directory
4023                              <1>
4024                              <1> loc_rmdir_cmd_failed:
4025                              <1>     ; 29/12/2017
4026 00008F8D 09C0                <1>     or      eax, eax ; EAX = 0 -> Directory not empty!
4027 00008F8F 7423                <1>     jz      short loc_rmdir_directory_not_empty
4028                              <1>
4029                              <1>     ; EAX > 0 -> Error code in AL (or AX or EAX)
4030                              <1>
4031 00008F91 833D[5E7E0100]01    <1>     cmp     dword [FAT_ClusterCounter], 1
4032                              <1>     ;jb     loc_file_rw_cmd_failed
4033                              <1>     ; 28/07/2022
4034 00008F98 7305                <1>     jnb     short loc_rmdir_failed
4035 00008F9A E999FDFFFF          <1>     jmp     loc_file_rw_cmd_failed
4036                              <1>
4037                              <1> loc_rmdir_failed:
4038 00008F9F F9                  <1>     stc
4039                              <1> loc_rmdir_cmd_return:
4040                              <1>     ; 01/03/2016
4041 00008FA0 9C                  <1>     pushf
4042                              <1>     ; ESI = Logical DOS Drive Description Table address
4043 00008FA1 66BB00FF            <1>     mov     bx, 0FF00h ; BH = FFh -> use ESI for Drive parameters
4044                              <1>                        ; BL = 0 -> Recalculate free cluster count
4045 00008FA5 50                  <1>     push    eax
4046 00008FA6 E8B6350000          <1>     call    calculate_fat_freespace
4047 00008FAB 58                  <1>     pop     eax
4048 00008FAC 9D                  <1>     popf
4049                              <1>     ;jc     loc_file_rw_cmd_failed
4050                              <1>     ;jmp    loc_file_rw_restore_retn
4051                              <1>     ; 28/07/2022
4052 00008FAD 73D5                <1>     jnc     short loc_rmdir_rw_restore_retn
4053 00008FAF E984FDFFFF          <1>     jmp     loc_file_rw_cmd_failed
4054                              <1>
4055                              <1> loc_rmdir_directory_not_empty:
4056 00008FB4 BE[1E330100]        <1>     mov     esi, Msg_Dir_Not_Empty
4057 00008FB9 E8ECDDFFFF          <1>     call    print_msg
4058                              <1>     ; 01/03/2016
4059 00008FBE A1[5E7E0100]        <1>     mov     eax, [FAT_ClusterCounter]
4060 00008FC3 09C0                <1>     or      eax, eax ; 0 ?
4061                              <1>     ;jz     loc_file_rw_restore_retn
4062                              <1>     ; 28/07/2022
4063 00008FC5 74BD                <1>     jz      short loc_rmdir_rw_restore_retn
4064                              <1>
4065                              <1>     ; ESI = Logical DOS Drive Description Table address
4066 00008FC7 66BB01FF            <1>     mov     bx, 0FF01h ; BH = FFh -> use ESI for Drive parameters
4067                              <1>                        ; BL = 1 -> add free clusters
4068 00008FCB E891350000          <1>     call    calculate_fat_freespace
4069 00008FD0 09C9                <1>     or      ecx, ecx
4070                              <1>     ;jz     loc_file_rw_restore_retn ; ecx = 0 -> OK
4071                              <1>     ;; ecx > 0 -> Error (Recalculation is needed)
```

```
4072                                    <1>     ;jmp    short loc_rmdir_cmd_return
4073                                    <1>     ; 28/07/2022
4074 00008FD2 75CC                      <1>     jnz     short loc_rmdir_cmd_return
4075 00008FD4 EBAE                      <1>     jmp     short loc_rmdir_rw_restore_retn
4076                                    <1>     ;jmp    loc_file_rw_restore_retn
4077                                    <1>
4078                                    <1> delete_sub_directory:
4079                                    <1>     ; 19/12/2025
4080                                    <1>     ; 17/07/2025 (TRDOS 386 Kernel v2.0.10)
4081                                    <1>     ; 28/07/2022 (TRDOS 386 Kernel v2.0.5)
4082                                    <1>     ; 29/12/2017
4083                                    <1>     ; (moved here from 'delete_directory' for 'sysrmdir' )
4084                                    <1>
4085                                    <1>     ; EDI = Directory buffer entry offset/address
4086                                    <1>
4087                                    <1> loc_rmdir_delete_short_name_check_dir_empty:
4088 00008FD6 668B4714                  <1>     mov     ax, [edi+20] ; First Cluster High Word
4089 00008FDA C1E010                    <1>         shl eax, 16
4090 00008FDD 668B471A                  <1>     mov     ax, [edi+26] ; First Cluster Low Word
4091                                    <1>
4092                                    <1>     ;;;
4093                                    <1>     ; 17/07/2025
4094                                    <1>     ; (sure it is not the current directory)
4095 00008FE1 8A15[E8800100]            <1>     mov     dl, [rmdir_drv]
4096 00008FE7 3A15[42770100]            <1>     cmp     dl, [Current_Drv]
4097 00008FED 7508                      <1>     jne     short loc_rmdir_delete_short_name_check_dot
4098 00008FEF 3B05[E4800100]            <1>     cmp     eax, [rmdir_dir_fcluster]
4099 00008FF5 743F                      <1>     je      short loc_rmdir_permission_denied
4100                                    <1> loc_rmdir_delete_short_name_check_dot:
4101                                    <1>     ; (DOT and DOTDOT can not be deleted)
4102 00008FF7 803F2E                    <1>     cmp     byte [edi], '.'
4103 00008FFA 750F                      <1>     jne     short loc_rmdir_delete_short_name_skip_dot
4104 00008FFC 66813F2E20                <1>     cmp     word [edi], '. '
4105 00009001 7433                      <1>     je      short loc_rmdir_permission_denied
4106 00009003 813F2E2E2020              <1>     cmp     dword [edi], '..  '
4107 00009009 742B                      <1>     je      short loc_rmdir_permission_denied
4108                                    <1> loc_rmdir_delete_short_name_skip_dot:
4109                                    <1>     ;;;
4110                                    <1>
4111                                    <1>     ;mov    [DelFile_FCluster], eax
4112                                    <1>
4113                                    <1>     ;;mov   bx, [DirBuff_EntryCounter]
4114                                    <1>     ;mov    bx, [FindFile_DirEntryNumber] ; 27/02/2016
4115                                    <1>     ;mov    [DelFile_EntryCounter], bx
4116                                    <1>
4117 0000900B 29DB                      <1>     sub     ebx, ebx
4118                                    <1>     ; 29/12/2017
4119 0000900D 891D[5E7E0100]            <1>     mov     [FAT_ClusterCounter], ebx ; 0 ; Reset
4120                                    <1>
4121 00009013 8A3D[E27F0100]            <1>     mov     bh, [FindFile_Drv]
4122 00009019 BE00010900                <1>     mov     esi, Logical_DOSDisks
4123 0000901E 01DE                      <1>     add     esi, ebx
4124                                    <1>
4125                                    <1>     ; 19/12/2025
4126                                    <1>     ;cmp    word [edi+DirEntry_NTRes], 01A1h
4127                                    <1>     ;je     short loc_rmdir_check_fs_directory
4128                                    <1>
4129                                    <1>     ;cmp    byte [esi+LD_FATType], 1
4130                                    <1>     ;jb     short loc_rmdir_get__last_cluster_0
4131                                    <1>
4132                                    <1>     ; 29/12/2017
4133 00009020 83F802                    <1>     cmp     eax, 2
4134 00009023 7306                      <1>     jnb     short loc_rmdir_get_last_cluster_1
4135                                    <1>     ; eax < 2
4136                                    <1> loc_rmdir_get_last_cluster_0:
4137                                    <1>     ;mov    eax, ERR_INV_FORMAT ; invalid format!
4138 00009025 B813000000                <1>     mov     eax, ERR_NOT_DIR ; not a valid directory!
4139                                    <1>     ;stc
4140 0000902A C3                        <1>     retn
4141                                    <1>
4142                                    <1> loc_rmdir_get_last_cluster_1:
4143 0000902B 807E0303                  <1>     cmp     byte [esi+LD_FATType], 3 ; FAT32
4144 0000902F 750C                      <1>     jne     short loc_rmdir_get_last_cluster_2
4145                                    <1>
4146                                    <1>     ; is it root directory ?
4147 00009031 3B4632                    <1>     cmp     eax, [esi+LD_BPB+BPB_RootClus]
4148 00009034 7507                      <1>     jne     short loc_rmdir_get_last_cluster_2
4149                                    <1>
4150                                    <1>     ; root directory can not be deleted !!
4151                                    <1> loc_rmdir_permission_denied:
4152 00009036 B80B000000                <1>     mov     eax, ERR_PERM_DENIED ; permission denied!
4153 0000903B F9                        <1>     stc
4154 0000903C C3                        <1>     retn
4155                                    <1>
4156                                    <1> loc_rmdir_get_last_cluster_2:
4157                                    <1>     ; 29/12/2017
4158 0000903D A3[A4800100]              <1>     mov     [DelFile_FCluster], eax
4159                                    <1>
4160                                    <1>     ;mov    dx, [DirBuff_EntryCounter]
4161 00009042 668B15[5C800100]          <1>     mov     dx, [FindFile_DirEntryNumber] ; 27/02/2016
4162 00009049 668915[A8800100]          <1>     mov     [DelFile_EntryCounter], dx
4163                                    <1>
4164 00009050 8B15[6E7E0100]            <1>     mov     edx, [DirBuff_Cluster]
4165 00009056 8915[D8800100]            <1>     mov     [RmDir_ParentDirCluster], edx
4166                                    <1>
4167 0000905C 893D[D4800100]            <1>     mov     [RmDir_DirEntryOffset], edi
4168                                    <1>
4169                                    <1>     ; 01/03/2016
4170                                    <1>     ;mov    dword [FAT_ClusterCounter], 0 ; Reset
4171                                    <1>
4172                                    <1> loc_rmdir_get_last_cluster_3:
4173 00009062 E878360000                <1>     call    get_last_cluster
4174                                    <1>         ;jc loc_rmdir_cmd_failed
4175 00009067 7211                      <1>     jc      short loc_delete_sub_dir_retn ; 29/12/2017
4176                                    <1>
4177 00009069 3B05[A4800100]            <1>     cmp     eax, [DelFile_FCluster]
4178 0000906F 750A                      <1>     jne     short loc_rmdir_multi_dir_clusters
4179                                    <1>
4180 00009071 C605[D3800100]00          <1>     mov     byte [RmDir_MultiClusters], 0
4181 00009078 EB08                      <1>     jmp     short pass_rmdir_multi_dir_clusters
4182                                    <1>
4183                                    <1> ; 19/12/2025
4184                                    <1> %if 0
4185                                    <1>
4186                                    <1> loc_rmdir_check_fs_directory:
4187                                    <1>     ; 29/12/2017
4188                                    <1>     cmp     byte [esi+LD_FSType], 0A1h
4189                                    <1>     jne     short loc_rmdir_permission_denied
4190                                    <1>
4191                                    <1> loc_rmdir_delete_fs_directory:
4192                                    <1>     call    delete_fs_directory
4193                                    <1>     ;jnc    loc_print_deleted_message
4194                                    <1>     jnc     short loc_delete_sub_dir_retn ; 29/12/2017
4195                                    <1>
```

```
                                <1>     ; EAX=0 -> Directory not empty !
                                <1>     ; EAX>0 -> Disk r/w error or another (misc) error
                                <1>
                                <1>     ;or     eax, eax
                                <1>     ;jz     loc_rmdir_directory_not_empty_2
                                <1>     ;;stc
                                <1>     ;;jmp   loc_file_rw_cmd_failed
                                <1>
                                <1> %endif
                                <1>
                                <1> loc_delete_sub_dir_retn:
0000907A C3                     <1>     retn
                                <1>
                                <1> loc_rmdir_multi_dir_clusters:
0000907B C605[D3800100]01       <1>     mov     byte [RmDir_MultiClusters], 1
                                <1>
                                <1> pass_rmdir_multi_dir_clusters:
00009082 A3[DC800100]           <1>     mov     [RmDir_DirLastCluster], eax
00009087 890D[E0800100]         <1>     mov     [RmDir_PreviousCluster], ecx
                                <1>
                                <1> loc_rmdir_load_fat_sub_directory:
0000908D E8A3300000             <1>     call    load_FAT_sub_directory
                                <1>     ;jc     loc_rmdir_cmd_failed
00009092 72E6                   <1>     jc      short loc_delete_sub_dir_retn
                                <1>
                                <1> loc_rmdir_find_last_dir_entry:
00009094 56                     <1>     push    esi
00009095 BE[C67F0100]           <1>     mov     esi, Dir_File_Name
0000909A C6062A                 <1>     mov     byte [esi], '*'
0000909D C646082A               <1>     mov     byte [esi+8], '*'
000090A1 31DB                   <1>     xor     ebx, ebx ; Entry offset  = 0
                                <1> loc_rmdir_find_last_dir_entry_next:
                                <1>     ;mov    ax, 0800h ; Except volume/long names
                                <1>     ;xor    cx, cx ; 0 = Find a valid file or dir name
                                <1>     ; 28/07/2022
000090A3 31C0                   <1>     xor     eax, eax
000090A5 B408                   <1>     mov     ah, 8
                                <1>     ; eax = 0800h
000090A7 31C9                   <1>     xor     ecx, ecx ; 0
000090A9 E825160000             <1>     call    find_directory_entry
000090AE 7225                   <1>     jc      short loc_rmdir_empty_dir_cluster
000090B0 83FB01                 <1>     cmp     ebx, 1
000090B3 771B                   <1>     ja      short loc_rmdir_directory_not_empty_1
                                <1> loc_rmdir_dot_entry_check:
000090B5 80FD2E                 <1>     cmp     ch, '.' ; The first char of the dir entry
000090B8 7516                   <1>     jne     short loc_rmdir_directory_not_empty_1
000090BA 08DB                   <1>     or      bl, bl
000090BC 7506                   <1>     jnz     short loc_rmdir_dotdot_entry_check
000090BE 807F0120               <1>     cmp     byte [edi+1], 20h
000090C2 EB06                   <1>     jmp     short pass_rmdir_dot_entry_check
                                <1>
                                <1> loc_rmdir_dotdot_entry_check:
000090C4 66817F012E20           <1>     cmp     word [edi+1], '. '
                                <1> pass_rmdir_dot_entry_check:
000090CA 7504                   <1>     jne     short loc_rmdir_directory_not_empty_1
000090CC FEC3                   <1>     inc     bl
000090CE EBD3                   <1>     jmp     short loc_rmdir_find_last_dir_entry_next
                                <1>
                                <1> loc_rmdir_directory_not_empty_1:
000090D0 58                     <1>     pop     eax ; pushed esi
000090D1 31C0                   <1>     xor     eax, eax ; 0
                                <1> loc_rmdir_directory_not_empty_2:
                                <1> loc_delete_sub_dir_stc_retn:
000090D3 F9                     <1>     stc
000090D4 C3                     <1>     retn
                                <1>
                                <1> loc_rmdir_empty_dir_cluster:
000090D5 5E                     <1>     pop     esi
                                <1>
                                <1> loc_rmdir_set_prev_cluster_dir_last_cluster:
000090D6 803D[D3800100]00       <1>     cmp     byte [RmDir_MultiClusters], 0
000090DD 7613                   <1>     jna     short loc_rmdir_unlink_dir_last_cluster
                                <1>
000090DF A1[E0800100]           <1>     mov     eax, [RmDir_PreviousCluster]
                                <1>     ;xor    ecx, ecx
000090E4 49                     <1>     dec     ecx ; FFFFFFFFh
000090E5 E876310000             <1>     call    update_cluster
000090EA 7306                   <1>     jnc     short loc_rmdir_unlink_dir_last_cluster
                                <1>
                                <1>     ; 01/03/2016
                                <1>     ;cmp    eax, 1  ; eax = 0 -> end of cluster chain
                                <1>     ;cmc
                                <1>     ;jc     short loc_rmdir_cmd_failed
                                <1>     ;jmp    short loc_rmdir_save_fat_buffer
                                <1>     ; 29/12/2017
000090EC 21C0                   <1>     and     eax, eax
000090EE 75E3                   <1>     jnz     short loc_delete_sub_dir_stc_retn
000090F0 EB12                   <1>     jmp     short loc_rmdir_save_fat_buffer
                                <1>
                                <1> loc_rmdir_unlink_dir_last_cluster:
000090F2 A1[DC800100]           <1>     mov     eax, [RmDir_DirLastCluster]
000090F7 31C9                   <1>     xor     ecx, ecx ; 0
000090F9 E862310000             <1>     call    update_cluster
000090FE 7327                   <1>     jnc     short loc_rmdir_unlink_stc_retn_0Bh
                                <1>     ; Because of it is the last cluster
                                <1>     ; 'update_cluster' must return with eocc error
00009100 09C0                   <1>     or      eax, eax
                                <1>     ;jz     short loc_rmdir_save_fat_buffer ; eocc
                                <1>     ;stc
                                <1>         ;jmp       short loc_rmdir_cmd_failed
                                <1>     ; 29/12/2017
00009102 75CF                   <1>     jnz     short loc_delete_sub_dir_stc_retn
                                <1>
                                <1> loc_rmdir_save_fat_buffer:
00009104 803D[567E0100]02       <1>     cmp     byte [FAT_BuffValidData], 2
0000910B 7527                   <1>     jne     short loc_rmdir_calculate_FAT_freespace
0000910D E8C2330000             <1>     call    save_fat_buffer
                                <1>     ;jc     short loc_rmdir_cmd_failed
                                <1>     ; 29/12/2017
00009112 7219                   <1>     jc      short loc_rmdir_unlink_error_retn
                                <1>
                                <1>     ; 01/03/2016
00009114 803D[D3800100]00       <1>     cmp     byte [RmDir_MultiClusters], 0
0000911B 7617                   <1>     jna     short loc_rmdir_calculate_FAT_freespace
                                <1>
0000911D A1[A4800100]           <1>     mov     eax, [DelFile_FCluster]
00009122 E93BFFFFFF             <1>         jmp     loc_rmdir_get_last_cluster_3
                                <1>
                                <1> loc_rmdir_unlink_stc_retn_0Bh:
                                <1>     ; 15/10/2016 (0Bh -> 28)
00009127 B81C000000             <1>     mov     eax, ERR_INV_FORMAT ; 28 = Invalid format
                                <1> loc_rmdir_unlink_stc_retn:
0000912C F9                     <1>     stc
                                <1> loc_rmdir_unlink_error_retn:
```

```
4320 0000912D C3                <1>     retn
4321                            <1>
4322                            <1> loc_rmdir_delete_short_name_invalid_data:
4323                            <1>     ;mov    eax, 29 ; Invalid data (15/10/2016)
4324                            <1>     ; 28/07/2022
4325 0000912E 29C0              <1>     sub     eax, eax
4326 00009130 B01D              <1>     mov     al, 29
4327                            <1>     ;stc
4328                            <1>         ;jmp       loc_rmdir_cmd_failed
4329                            <1>     ; 29/12/2017
4330 00009132 EBF8              <1>     jmp     short loc_rmdir_unlink_stc_retn
4331                            <1>
4332                            <1> loc_rmdir_calculate_FAT_freespace:
4333                            <1>     ;mov    eax, [FAT_ClusterCounter]
4334                            <1>     ; 29/12/2017
4335 00009134 29C0              <1>     sub     eax, eax ; 0
4336 00009136 8705[5E7E0100]    <1>     xchg    eax, [FAT_ClusterCounter]
4337                            <1>     ;
4338 0000913C 66BB01FF          <1>     mov     bx, 0FF01h
4339                            <1>     ; BL = 1 -> Add EAX to free space count
4340                            <1>     ; BH = FFh ->
4341                            <1>     ; ESI = Logical DOS Drive Description Table address
4342 00009140 E81C340000        <1>     call    calculate_fat_freespace
4343                            <1>
4344 00009145 21C9              <1>     and     ecx, ecx ; ecx = 0 -> valid free sector count
4345 00009147 7409              <1>     jz      short loc_rmdir_delete_short_name_continue
4346                            <1>
4347                            <1> loc_rmdir_recalculate_FAT_freespace:
4348 00009149 66BB00FF          <1>         mov     bx, 0FF00h ; BL = 0 -> Recalculate free space
4349 0000914D E80F340000        <1>     call    calculate_fat_freespace
4350                            <1>
4351                            <1> loc_rmdir_delete_short_name_continue:
4352 00009152 A1[D8800100]      <1>     mov     eax, [RmDir_ParentDirCluster]
4353 00009157 83F802            <1>     cmp     eax, 2
4354 0000915A 7309              <1>     jnb     short loc_rmdir_del_short_name_load_sub_dir
4355 0000915C E8562F0000        <1>     call    load_FAT_root_directory
4356                            <1>     ;jc     loc_file_rw_cmd_failed
4357                            <1>     ; 29/12/2017
4358 00009161 72CA              <1>     jc      short loc_rmdir_unlink_error_retn
4359 00009163 EB07              <1>     jmp     short loc_rmdir_del_short_name_ld_chk_fclust
4360                            <1>
4361                            <1> loc_rmdir_del_short_name_load_sub_dir:
4362 00009165 E8CB2F0000        <1>     call    load_FAT_sub_directory
4363                            <1>     ;jc     loc_file_rw_cmd_failed
4364                            <1>     ; 29/12/2017
4365 0000916A 72C1              <1>     jc      short loc_rmdir_unlink_error_retn
4366                            <1>
4367                            <1> loc_rmdir_del_short_name_ld_chk_fclust:
4368 0000916C 0FB73D[D4800100]  <1>     movzx   edi, word [RmDir_DirEntryOffset]
4369 00009173 81C700000800      <1>     add     edi, Directory_Buffer
4370                            <1>
4371 00009179 668B4714          <1>     mov     ax, [edi+20] ; First Cluster High Word
4372 0000917D C1E010            <1>     shl     eax, 16
4373 00009180 668B471A          <1>     mov     ax, [edi+26] ; First Cluster Low Word
4374                            <1>         ; Not necessary...
4375 00009184 3B05[A4800100]    <1>     cmp     eax, [DelFile_FCluster]
4376 0000918A 75A2              <1>     jne     short loc_rmdir_delete_short_name_invalid_data
4377                            <1>     ;
4378 0000918C C607E5            <1>     mov     byte [edi], 0E5h ; 'Deleted' sign
4379                            <1>     ; 27/02/2016
4380                            <1>     ; TRDOS v1 has a bug here! it does not set
4381                            <1>     ; 'DirBuff_ValidData' to 2; as result of this bug,
4382                            <1>     ; 'save_directory_buffer' would not save the change !
4383 0000918F C605[697E0100]02    <1>     mov     byte [DirBuff_ValidData], 2 ; change sign
4384                            <1>     ;
4385 00009196 E8E61B0000        <1>     call    save_directory_buffer
4386                            <1>     ;jc     loc_file_rw_cmd_failed
4387                            <1>     ; 29/12/2017
4388 0000919B 7290              <1>     jc      short loc_rmdir_unlink_error_retn
4389                            <1>
4390                            <1> loc_rmdir_del_long_name:
4391 0000919D 0FB615[AA800100]  <1>     movzx   edx, byte [DelFile_LNEL]
4392 000091A4 08D2              <1>     or      dl, dl
4393 000091A6 7410              <1>     jz      short loc_rmdir_update_parent_dir_lmdt
4394                            <1>
4395 000091A8 0FB705[A8800100]  <1>     movzx   eax, word [DelFile_EntryCounter]
4396 000091AF 29D0              <1>     sub     eax, edx
4397                            <1>     ; 29/12/2017
4398 000091B1 7205              <1>     jc      short loc_rmdir_update_parent_dir_lmdt
4399                            <1>
4400                            <1>     ; EAX = Directory Entry Number of the long name last entry
4401 000091B3 E8091D0000        <1>     call    delete_longname
4402                            <1>
4403                            <1> loc_rmdir_update_parent_dir_lmdt:
4404 000091B8 E84D1C0000        <1>     call    update_parent_dir_lmdt
4405                            <1>     ;jc     short loc_file_rw_cmd_failed
4406                            <1>     ; 29/12/2017
4407                            <1>     ;jc     short loc_rmdir_unlink_error_retn
4408                            <1>
4409                            <1> loc_delete_sub_directory_ok:
4410                            <1>     ; 29/12/2017
4411 000091BD 31C0              <1>     xor     eax, eax ;  0 ;  cf = 0
4412 000091BF C3                <1>     retn
4413                            <1>
4414                            <1> delete_file:
4415                            <1>     ; 28/07/2022 (TRDOS 386 Kernel v2.0.5)
4416                            <1>     ; 29/02/2016
4417                            <1>     ; 28/02/2016 (TRDOS 386 = TRDOS v2.0)
4418                            <1>     ; 09/08/2010 (CMD_INTR.ASM, 'cmp_cmd_del')
4419                            <1>     ; 28/02/2010
4420                            <1>
4421                            <1> get_delfile_fchar:
4422                            <1>     ; esi = file name
4423 000091C0 803E20            <1>     cmp     byte [esi], 20h
4424 000091C3 7701              <1>         ja  short loc_delfile_parse_path_name
4425                            <1>
4426                            <1> loc_delfile_nofilename_retn:
4427 000091C5 C3                <1>     retn
4428                            <1>
4429                            <1> loc_delfile_parse_path_name:
4430 000091C6 BF[E27F0100]      <1>     mov     edi, FindFile_Drv
4431 000091CB E89E170000        <1>     call    parse_path_name
4432                            <1>     ;jc     loc_cmd_failed
4433                            <1>     ; 28/07/2022
4434 000091D0 7305              <1>     jnc     short loc_delfile_check_filename_exists
4435                            <1> loc_delfile_failed:
4436 000091D2 E96DF2FFFF        <1>     jmp     loc_cmd_failed
4437                            <1>
4438                            <1> loc_delfile_check_filename_exists:
4439 000091D7 BE[24800100]      <1>     mov     esi, FindFile_Name
4440 000091DC 803E20            <1>     cmp     byte [esi], 20h
4441                            <1>     ;jna    loc_cmd_failed
4442                            <1>     ; 28/07/2022
4443 000091DF 76F1              <1>     jna     short loc_delfile_failed
```

```
4444 000091E1 8935[A0800100]      <1>     mov     [DelFile_FNPointer], esi
4445                              <1>
4446                              <1> loc_delfile_drv:
4447 000091E7 8A15[E27F0100]      <1>     mov     dl, [FindFile_Drv]
4448 000091ED 8A35[42770100]      <1>     mov     dh, [Current_Drv]
4449 000091F3 8835[9F7E0100]      <1>     mov     [RUN_CDRV], dh
4450 000091F9 38F2                <1>     cmp     dl, dh
4451 000091FB 7407                <1>     je      short loc_delfile_change_directory
4452                              <1>
4453 000091FD E8DEE4FFFF          <1>     call    change_current_drive
4454                              <1>     ;jc     loc_file_rw_cmd_failed
4455                              <1>     ; 28/07/2022
4456                              <1>     ;jnc    short loc_delfile_change_directory
4457                              <1>     ;jmp    loc_file_rw_cmd_failed
4458 00009202 721D                <1>     jc      short loc_delfile_chdrv_failed
4459                              <1>
4460                              <1> loc_delfile_change_directory:
4461 00009204 803D[E37F0100]20    <1>     cmp     byte [FindFile_Directory], 20h
4462 0000920B 7619                <1>     jna     short loc_delfile_find
4463                              <1>
4464 0000920D FE05[5D2E0100]      <1>     inc     byte [Restore_CDIR]
4465 00009213 BE[E37F0100]        <1>     mov     esi, FindFile_Directory
4466 00009218 30E4                <1>     xor     ah, ah ; CD_COMMAND sign -> 0
4467 0000921A E8BC110000          <1>     call    change_current_directory
4468                              <1>     ;jc     loc_file_rw_cmd_failed
4469                              <1>     ; 28/07/2022
4470 0000921F 7305                <1>     jnc     short loc_delfile_chdir_ok
4471                              <1> loc_delfile_chdrv_failed:
4472                              <1> loc_delfile_fff_failed:
4473 00009221 E912FBFFFF          <1>     jmp     loc_file_rw_cmd_failed
4474                              <1>
4475                              <1> loc_delfile_chdir_ok: ; 28/07/2022
4476                              <1>
4477                              <1> ;loc_delfile_change_prompt_dir_string:
4478                              <1>     ;call   change_prompt_dir_string
4479                              <1>
4480                              <1> loc_delfile_find:
4481                              <1>     ;mov    esi, FindFile_Name
4482 00009226 8B35[A0800100]      <1>     mov     esi, [DelFile_FNPointer]
4483 0000922C 66B80018            <1>     mov     ax, 1800h ; Except volume label and dirs
4484 00009230 E835F7FFFF          <1>     call    find_first_file
4485                              <1>     ;jc     loc_file_rw_cmd_failed
4486                              <1>     ; 28/07/2022
4487 00009235 72EA                <1>     jc      short loc_delfile_fff_failed
4488                              <1>
4489                              <1> loc_delfile_ambgfn_check:
4490 00009237 6621D2              <1>     and     dx, dx ; Ambiguous filename chars used sign (DX>0)
4491 0000923A 740A                <1>     jz      short loc_delfile_found
4492                              <1>
4493                              <1> loc_file_not_found:
4494                              <1>     ;mov    eax, 2 ; File not found sign
4495                              <1>     ; 28/07/2022
4496 0000923C 31C0                <1>     xor     eax, eax
4497 0000923E B002                <1>     mov     al, 2
4498 00009240 F9                  <1>     stc
4499 00009241 E9F2FAFFFF          <1>     jmp     loc_file_rw_cmd_failed
4500                              <1>
4501                              <1> loc_delfile_found:
4502 00009246 80E307              <1>     and     bl, 07h ; Attributes
4503                              <1>     ;jnz    loc_permission_denied
4504                              <1>     ; 28/07/2022
4505 00009249 7405                <1>     jz      short loc_delfile_attrb_ok
4506 0000924B E9F2FAFFFF          <1>     jmp     loc_permission_denied
4507                              <1>
4508                              <1> loc_delfile_attrb_ok: ; 28/07/2022
4509                              <1>
4510                              <1> ;loc_delfile_found_save_lnel:
4511                              <1> ;   mov     [DelFile_LNEL], bh ; Long name entry length (if > 0)
4512                              <1>
4513                              <1> loc_delfile_ask_for_delete:
4514 00009250 57                  <1>     push    edi ; * (29/02/2016)
4515                              <1>
4516 00009251 BE[35330100]        <1>     mov     esi, Msg_DoYouWantDelete
4517 00009256 E84FDBFFFF          <1>     call    print_msg
4518 0000925B 8B35[A0800100]      <1>     mov     esi, [DelFile_FNPointer]
4519 00009261 E844DBFFFF          <1>     call    print_msg
4520 00009266 BE[E9320100]        <1>     mov     esi, Msg_YesNo
4521 0000926B E83ADBFFFF          <1>     call    print_msg
4522                              <1>
4523                              <1> loc_delfile_ask_again:
4524 00009270 30E4                <1>     xor     ah, ah
4525 00009272 E8A57CFFFF          <1>     call    int16h
4526 00009277 3C1B                <1>     cmp     al, 1Bh
4527                              <1>     ;je     short loc_do_not_delete_file
4528 00009279 744C                <1>     je      short loc_delfile_y_n_escape ; 06/03/2016
4529 0000927B 24DF                <1>     and     al, 0DFh
4530 0000927D A2[F3320100]        <1>     mov     [Y_N_nextline], al
4531 00009282 3C59                <1>     cmp     al, 'Y'
4532 00009284 7404                <1>     je      short loc_yes_delete_file
4533 00009286 3C4E                <1>     cmp     al, 'N'
4534 00009288 75E6                <1>     jne     short loc_delfile_ask_again
4535                              <1>
4536                              <1> loc_do_not_delete_file:
4537                              <1> loc_yes_delete_file:
4538 0000928A E8DCFBFFFF          <1>     call    y_n_answer ; 29/12/2017
4539 0000928F 5F                  <1>     pop     edi ; * (29/02/2016)
4540                              <1>     ;cmp    al, 'Y' ; 'yes'
4541                              <1>     ;cmc
4542                              <1>         ;jnc       loc_file_rw_restore_retn
4543 00009290 3C4E                <1>     cmp     al, 'N' ; 'no'
4544                              <1>         ;je loc_file_rw_restore_retn
4545                              <1>     ; 28/07/2022
4546 00009292 7505                <1>     jne     short loc_delete_file
4547 00009294 E99FFAFFFF          <1>     jmp     loc_file_rw_restore_retn
4548                              <1>
4549                              <1> loc_delete_file:
4550 00009299 8A3D[E27F0100]      <1>     mov     bh, [FindFile_Drv]
4551                              <1>     ;mov    bl, [DelFile_LNEL]
4552 0000929F 8A1D[31800100]      <1>     mov     bl, [FindFile_LongNameEntryLength]
4553                              <1>     ;mov    cx, [DirBuff_EntryCounter]
4554 000092A5 668B0D[5C800100]    <1>     mov     cx, [FindFile_DirEntryNumber]
4555                              <1>     ; (*) EDI = Directory buffer entry offset/address
4556 000092AC E8E81D0000          <1>     call    remove_file ; (FILE.ASM, 'proc_delete_file')
4557                              <1>     ;jnc    loc_print_deleted_message
4558                              <1>     ; 28/07/2022
4559 000092B1 7205                <1>     jc      short loc_delete_file_err1
4560 000092B3 E976FAFFFF          <1>     jmp     loc_print_deleted_message
4561                              <1>
4562                              <1> loc_delete_file_err1:
4563                              <1>     ;cmp    al, 05h
4564 000092B8 3C0B                <1>     cmp     al, ERR_PERM_DENIED  ; 29/12/2017 (5 -> 11)
4565                              <1>     ;je     loc_permission_denied
4566                              <1>     ; 28/07/2022
4567 000092BA 7505                <1>     jne     short loc_delete_file_err2
```

```
4568 000092BC E981FAFFFF          <1>     jmp     loc_permission_denied
4569                              <1> loc_delete_file_err2:
4570 000092C1 F9                  <1>     stc
4571 000092C2 E971FAFFFF          <1>     jmp     loc_file_rw_cmd_failed
4572                              <1>
4573                              <1> loc_delfile_y_n_escape:
4574 000092C7 B04E                <1>     mov     al, 'N' ; 'no'
4575 000092C9 EBBF                <1>     jmp     short loc_do_not_delete_file
4576                              <1>
4577                              <1> set_file_attributes:
4578                              <1>     ; 19/12/2025 (TRDOS 386 v2.0.10)
4579                              <1>     ; 26/09/2024 (TRDOS 386 v2.0.9)
4580                              <1>     ; 28/07/2022 (TRDOS 386 Kernel v2.0.5)
4581                              <1>     ; 06/03/2016
4582                              <1>     ; 04/03/2016 (TRDOS 386 = TRDOS v2.0)
4583                              <1>     ; 10/07/2010 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_attrib')
4584                              <1>     ; 23/05/2010
4585                              <1>     ; 17/12/2000 (P2000.ASM)
4586                              <1>
4587                              <1>     ; esi = file or directory name
4588                              <1>     ;xor   ax, ax
4589                              <1>     ; 28/07/2022
4590 000092CB 31C0                <1>     xor     eax, eax
4591 000092CD 66A3[86330100]      <1>     mov     [Attr_Chars], ax
4592 000092D3 A2[FC800100]        <1>     mov     [Attributes], al
4593                              <1>
4594                              <1> get_attrib_fchar:
4595                              <1>     ; esi = file name
4596 000092D8 8A06                <1>     mov     al, [esi]
4597 000092DA 3C20                <1>     cmp     al, 20h
4598 000092DC 7621                <1>     jna     short loc_attr_file_nofilename_retn
4599                              <1>
4600                              <1> loc_scan_attrib_params:
4601 000092DE 3C2D                <1>     cmp     al, '-'
4602                              <1>     ;ja     loc_attr_file_parse_path_name
4603                              <1>     ;je     short loc_attr_space
4604                              <1>     ; 28/07/2022
4605 000092E0 7207                <1>     jb      short loc_sfa_1
4606 000092E2 740E                <1>     je      short loc_attr_space
4607 000092E4 E911010000          <1>     jmp     loc_attr_file_parse_path_name
4608                              <1>
4609                              <1> loc_sfa_1:
4610                              <1>     ; 28/07/2022
4611 000092E9 3C2B                <1>     cmp     al, '+'
4612                              <1>     ;jne    loc_cmd_failed
4613 000092EB 7405                <1>     je      short loc_attr_space
4614                              <1> loc_sfa_2:
4615 000092ED E952F1FFFF          <1>     jmp     loc_cmd_failed
4616                              <1> loc_attr_space:
4617 000092F2 8A6601              <1>     mov     ah, [esi+1]
4618 000092F5 80FC20              <1>     cmp     ah, 20h
4619 000092F8 7706                <1>     ja      short pass_attr_space
4620                              <1>     ;jb     loc_cmd_failed
4621                              <1>     ; 28/07/2022
4622 000092FA 72F1                <1>     jb      short loc_sfa_2
4623 000092FC 46                  <1>     inc     esi
4624 000092FD EBF3                <1>     jmp     short loc_attr_space
4625                              <1>
4626                              <1> loc_attr_file_nofilename_retn:
4627 000092FF C3                  <1>     retn
4628                              <1>
4629                              <1> pass_attr_space:
4630 00009300 80E4DF              <1>     and     ah, 0DFh
4631 00009303 80FC53              <1>     cmp     ah, 'S'
4632                              <1>     ;ja     loc_cmd_failed
4633                              <1>     ; 28/07/2022
4634 00009306 77E5                <1>     ja      short loc_sfa_2
4635 00009308 7204                <1>     jb      short pass_attr_system
4636 0000930A B404                <1>     mov     ah, 04h ; System
4637 0000930C EB1D                <1>     jmp     short pass_attr_archive
4638                              <1>
4639                              <1> pass_attr_system:
4640 0000930E 80FC48              <1>     cmp     ah, 'H'
4641 00009311 7706                <1>     ja      short pass_attr_hidden
4642 00009313 720F                <1>     jb      short pass_attr_read_only
4643 00009315 B402                <1>     mov     ah, 02h ; Hidden
4644 00009317 EB12                <1>     jmp     short pass_attr_archive
4645                              <1>
4646                              <1> pass_attr_hidden:
4647 00009319 80FC52              <1>     cmp     ah, 'R'
4648                              <1>     ;ja     loc_cmd_failed
4649                              <1>     ; 28/07/2022
4650 0000931C 77CF                <1>     ja      short loc_sfa_2
4651 0000931E 7204                <1>     jb      short pass_attr_read_only ; Read only
4652 00009320 B401                <1>     mov     ah, 01h
4653 00009322 EB07                <1>     jmp     short pass_attr_archive
4654                              <1>
4655                              <1> pass_attr_read_only:
4656 00009324 80FC41              <1>     cmp     ah, 'A'
4657 00009327 753B                <1>     jne     short loc_chk_attr_enter
4658 00009329 B420                <1>     mov     ah, 20h ; Archive
4659                              <1>
4660                              <1> pass_attr_archive:
4661 0000932B 3C2D                <1>     cmp     al, '-'
4662 0000932D 7508                <1>     jne     short pass_reducing_attributes
4663 0000932F 0825[86330100]      <1>     or      [Attr_Chars], ah
4664 00009335 EB06                <1>     jmp     short loc_change_attributes_inc
4665                              <1>
4666                              <1> pass_reducing_attributes:
4667 00009337 0825[87330100]      <1>     or      [Attr_Chars+1], ah
4668                              <1>
4669                              <1> loc_change_attributes_inc:
4670 0000933D 46                  <1>     inc     esi
4671 0000933E 8A6601              <1>     mov     ah, [esi+1]
4672 00009341 80FC20              <1>     cmp     ah, 20h
4673 00009344 7228                <1>     jb      short pass_change_attr
4674 00009346 74F5                <1>     je      short loc_change_attributes_inc
4675 00009348 80FC2D              <1>     cmp     ah, '-'
4676 0000934B 770D                <1>     ja      short loc_chk_next_attr_char1
4677 0000934D 7405                <1>     je      short loc_chk_next_attr_char0
4678 0000934F 80FC2B              <1>     cmp     ah, '+'
4679 00009352 7506                <1>     jne     short loc_chk_next_attr_char1
4680                              <1>
4681                              <1> loc_chk_next_attr_char0:
4682 00009354 46                  <1>     inc     esi
4683 00009355 668B06              <1>     mov     ax, [esi]
4684 00009358 EBA6                <1>     jmp     short pass_attr_space
4685                              <1>
4686                              <1> loc_chk_next_attr_char1:
4687 0000935A 803E2D              <1>     cmp     byte [esi], '-'
4688 0000935D 77A1                <1>     ja      short pass_attr_space
4689 0000935F E989000000          <1>         jmp     loc_attr_file_check_fname_fchar
4690                              <1>
4691                              <1> loc_chk_attr_enter:
```

```
4692 00009364 80FC0D              <1>     cmp     ah, 0Dh
4693                              <1>     ;jne    loc_cmd_failed
4694                              <1>     ; 28/07/202
4695 00009367 7405                <1>     je      short pass_change_attr
4696 00009369 E9D6F0FFFF          <1>     jmp     loc_cmd_failed
4697                              <1>
4698                              <1> pass_change_attr:
4699 0000936E A0[86330100]        <1>     mov     al, [Attr_Chars]
4700 00009373 F6D0                <1>     not     al
4701 00009375 2005[FC800100]      <1>     and     [Attributes], al
4702 0000937B A0[87330100]        <1>     mov     al, [Attr_Chars+1]
4703 00009380 0805[FC800100]      <1>     or      [Attributes], al
4704                              <1>
4705                              <1> loc_show_attributes:
4706 00009386 BE[E7380100]        <1>     mov     esi, nextline
4707 0000938B E81ADAFFFF          <1>     call    print_msg
4708                              <1>
4709                              <1> loc_show_attributes_no_nextline:
4710 00009390 C705[86330100]4E4F- <1>     mov     dword [Attr_Chars], 'NORM'
4710 00009398 524D                <1>
4711 0000939A 66C705[8A330100]41- <1>     mov     word [Attr_Chars+4], 'AL'
4711 000093A2 4C                  <1>
4712 000093A3 BE[86330100]        <1>     mov     esi, Attr_Chars
4713 000093A8 A0[FC800100]        <1>     mov     al, [Attributes]
4714 000093AD A804                <1>     test    al, 04h
4715 000093AF 7406                <1>     jz      short pass_put_attr_s
4716 000093B1 66C7065300          <1>     mov     word [esi], 0053h     ; S
4717 000093B6 46                  <1>     inc     esi
4718                              <1>
4719                              <1> pass_put_attr_s:
4720 000093B7 A802                <1>     test    al, 02h
4721 000093B9 7406                <1>     jz      short pass_put_attr_h
4722 000093BB 66C7064800          <1>     mov     word [esi], 0048h     ; H
4723 000093C0 46                  <1>     inc     esi
4724                              <1>
4725                              <1> pass_put_attr_h:
4726 000093C1 A801                <1>     test    al, 01h
4727 000093C3 7406                <1>     jz      short pass_put_attr_r
4728 000093C5 66C7065200          <1>     mov     word [esi], 0052h     ; R
4729 000093CA 46                  <1>     inc     esi
4730                              <1>
4731                              <1> pass_put_attr_r:
4732 000093CB 3C20                <1>     cmp     al, 20h
4733 000093CD 7205                <1>     jb      short pass_put_attr_a
4734 000093CF 66C7064100          <1>     mov     word [esi], 0041h     ; A
4735                              <1>
4736                              <1> pass_put_attr_a:
4737 000093D4 BE[79330100]        <1>     mov     esi, Str_Attributes
4738 000093D9 E8CCD9FFFF          <1>     call    print_msg
4739 000093DE BE[E7380100]        <1>     mov     esi, nextline
4740 000093E3 E8C2D9FFFF          <1>     call    print_msg
4741 000093E8 E94BF9FFFF          <1>     jmp     loc_file_rw_restore_retn
4742                              <1>
4743                              <1> loc_attr_file_check_fname_fchar:
4744 000093ED 46                  <1>     inc     esi
4745 000093EE 803E20              <1>     cmp     byte [esi], 20h
4746 000093F1 74FA                <1>     je      short loc_attr_file_check_fname_fchar
4747                              <1>         ;jb pass_change_attr
4748                              <1>     ; 28/07/2022
4749 000093F3 7705                <1>     ja      short loc_attr_file_parse_path_name
4750 000093F5 E974FFFFFF          <1>     jmp     pass_change_attr
4751                              <1>
4752                              <1> loc_attr_file_parse_path_name:
4753 000093FA BF[E27F0100]        <1>     mov     edi, FindFile_Drv
4754 000093FF E86A150000          <1>     call    parse_path_name
4755                              <1>     ;jc     loc_cmd_failed
4756                              <1>     ; 28/07/2022
4757 00009404 7305                <1>     jnc     short loc_attr_file_check_filename_exists
4758                              <1> loc_sfa_3:
4759 00009406 E939F0FFFF          <1>     jmp     loc_cmd_failed
4760                              <1>
4761                              <1> loc_attr_file_check_filename_exists:
4762 0000940B BE[24800100]        <1>     mov     esi, FindFile_Name
4763 00009410 803E20              <1>     cmp     byte [esi], 20h
4764                              <1>     ;jna    loc_cmd_failed
4765                              <1>     ; 28/07/2022
4766 00009413 76F1                <1>     jna     short loc_sfa_3
4767 00009415 8935[A0800100]      <1>     mov     [DelFile_FNPointer], esi
4768                              <1>
4769                              <1> loc_attr_file_drv:
4770 0000941B 8A35[42770100]      <1>     mov     dh, [Current_Drv]
4771 00009421 8835[9F7E0100]      <1>     mov     [RUN_CDRV], dh
4772                              <1>
4773 00009427 8A15[E27F0100]      <1>     mov     dl, [FindFile_Drv]
4774 0000942D 38F2                <1>     cmp     dl, dh
4775 0000942F 7407                <1>     je      short loc_attr_file_change_directory
4776                              <1>
4777 00009431 E8AAE2FFFF          <1>     call    change_current_drive
4778                              <1>     ;jc     loc_file_rw_cmd_failed
4779                              <1>     ; 28/07/2022
4780 00009436 722E                <1>     jc      short loc_sfa_4
4781                              <1>
4782                              <1> loc_attr_file_change_directory:
4783 00009438 803D[E37F0100]20    <1>         cmp     byte [FindFile_Directory], 20h
4784 0000943F 7614                <1>     jna     short loc_attr_file_find
4785                              <1>
4786 00009441 FE05[5D2E0100]      <1>     inc     byte [Restore_CDIR]
4787                              <1>
4788 00009447 BE[E37F0100]        <1>     mov     esi, FindFile_Directory
4789 0000944C 30E4                <1>     xor     ah, ah ; CD_COMMAND sign -> 0
4790 0000944E E8880F0000          <1>     call    change_current_directory
4791                              <1>     ;jc     loc_file_rw_cmd_failed
4792                              <1>     ; 28/07/2022
4793 00009453 7211                <1>     jc      short loc_sfa_4
4794                              <1>
4795                              <1> ;loc_attr_file_change_prompt_dir_string:
4796                              <1>     ;call   change_prompt_dir_string
4797                              <1>
4798                              <1> loc_attr_file_find:
4799                              <1>     ;mov    esi, FindFile_Name
4800 00009455 8B35[A0800100]      <1>     mov     esi, [DelFile_FNPointer]
4801 0000945B 66B80008            <1>     mov     ax, 0800h ; Except volume labels
4802 0000945F E806F5FFFF          <1>     call    find_first_file
4803                              <1>     ;jc     loc_file_rw_cmd_failed
4804                              <1>     ; 28/07/2022
4805 00009464 7305                <1>     jnc     short loc_attr_file_ambgfn_check
4806                              <1> loc_sfa_4:
4807 00009466 E9CDF8FFFF          <1>     jmp     loc_file_rw_cmd_failed
4808                              <1>
4809                              <1> loc_attr_file_ambgfn_check:
4810 0000946B 6609D2              <1>     or      dx, dx ; Ambiguous filename chars used sign (DX>0)
4811                              <1>     ;       (Note: It was BX in TRDOS v1)
4812                              <1>     ;;jz    short loc_attr_file_found
4813                              <1>     ;jnz    loc_file_not_found ; 06/03/2016
```

```
                                <1>     ; 28/07/2022
0000946E 7405                   <1>     jz      short loc_attr_file_found
00009470 E9C7FDFFFF             <1>     jmp     loc_file_not_found
                                <1>
                                <1>     ;mov    eax, 2 ; File not found sign
                                <1>     ;stc
                                <1>     ;jmp    loc_file_rw_cmd_failed
                                <1>
                                <1> loc_attr_file_found:
                                <1>     ; EDI = Directory buffer entry offset/address
                                <1>     ; BL = File (or Directory) Attributes
                                <1>     ;       (Note: It was 'CL' in TRDOS v1)
                                <1>     ; mov  bl, [EDI+0Bh]
                                <1>
00009475 66833D[86330100]00     <1>     cmp     word [Attr_Chars], 0
0000947D 770B                   <1>     ja      short loc_attr_file_change_attributes
0000947F 881D[FC800100]         <1>     mov     [Attributes], bl
00009485 E9FCFEFFFF             <1>     jmp     loc_show_attributes
                                <1>
                                <1> loc_attr_file_change_attributes:
0000948A A0[86330100]           <1>     mov     al, [Attr_Chars]
0000948F F6D0                   <1>     not     al
00009491 20C3                   <1>     and     bl, al
00009493 A0[87330100]           <1>     mov     al, [Attr_Chars+1]
00009498 08C3                   <1>     or      bl, al
                                <1>
                                <1>     ; 19/12/2025
                                <1>     ;cmp    word [edi+DirEntry_NTRes], 01A1h ; Singlix FS
                                <1>     ;je     short loc_attr_file_fs_check
                                <1>
0000949A 881D[FC800100]         <1>     mov     [Attributes], bl
000094A0 885F0B                 <1>     mov     [edi+0Bh], bl  ; Attributes (New!)
                                <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>
000094A3 C605[697E0100]02       <1>     mov     byte [DirBuff_ValidData], 2
                                <1>
000094AA E8D2180000             <1>     call    save_directory_buffer
                                <1>     ;jc     loc_file_rw_cmd_failed
                                <1>     ;jmp    short loc_print_attr_changed_message
                                <1>     ; 28/07/2022
000094AF 7305                   <1>     jnc     short loc_print_attr_changed_message
                                <1> loc_sfa_5:
000094B1 E982F8FFFF             <1>     jmp     loc_file_rw_cmd_failed
                                <1>
                                <1> ; 19/12/2025
                                <1> %if 0
                                <1> loc_attr_file_fs_check:
                                <1>     sub     eax, eax
                                <1>         mov     ah, [DirBuff_DRV]
                                <1>     ; 26/09/2024 (BugFix)
                                <1>     sub     ah, 'A'
                                <1>     mov     esi, Logical_DOSDisks
                                <1>         add     esi, eax
                                <1>         cmp     byte [esi+LD_FSType], 0A1h
                                <1>     jnc     short loc_attr_file_change_fs_file_attributes
                                <1>     ; 29/12/2017 (0Dh -> 29)
                                <1>     mov     ax, 29 ; Invalid Data
                                <1>     jmp     loc_file_rw_cmd_failed
                                <1>
                                <1> loc_attr_file_change_fs_file_attributes:
                                <1>     ; BL = New MS-DOS File Attributes
                                <1>     mov     al, bl ; File/Directory Attributes
                                <1>     xor     ah, ah ; Attributes in MS-DOS format sign
                                <1>     call    change_fs_file_attributes
                                <1>     ;jc     loc_file_rw_cmd_failed
                                <1>     ; 28/07/2022
                                <1>     jc      short loc_sfa_5
                                <1>
                                <1>     mov     [Attributes], bl
                                <1> %endif
                                <1>
                                <1> loc_print_attr_changed_message:
000094B6 BE[74330100]           <1>     mov     esi, Msg_New
000094BB E8EAD8FFFF             <1>     call    print_msg
000094C0 E9CBFEFFFF             <1>     jmp     loc_show_attributes_no_nextline
                                <1>
                                <1> rename_file:
                                <1>     ; 25/07/2025 (BugFix)
                                <1>     ; 28/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <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
000094C5 803E20                 <1>     cmp     byte [esi], 20h
000094C8 7613                   <1>         jna short loc_rename_nofilename_retn
                                <1>
000094CA 8935[24810100]         <1>     mov     [SourceFilePath], esi
                                <1>
                                <1> rename_scan_source_file:
000094D0 46                     <1>     inc     esi
000094D1 803E20                 <1>     cmp     byte [esi], 20h
000094D4 7408                   <1>     je      short rename_scan_destination_file_1
                                <1>     ;;jb    short loc_rename_nofilename_retn
                                <1>     ;jb     loc_cmd_failed
                                <1>     ;jmp    short rename_scan_source_file
                                <1>     ; 28/07/2022
000094D6 77F8                   <1>     ja      short rename_scan_source_file
                                <1> loc_rename_failed:
000094D8 E967EFFFFF             <1>     jmp     loc_cmd_failed
                                <1>
                                <1> loc_rename_nofilename_retn: ; 08/03/2016
000094DD C3                     <1>     retn
                                <1>
                                <1> rename_scan_destination_file_1:
000094DE C60600                 <1>     mov     byte [esi], 0
                                <1>
                                <1> rename_scan_destination_file_2:
000094E1 46                     <1>     inc     esi
000094E2 803E20                 <1>     cmp     byte [esi], 20h
000094E5 74FA                   <1>     je      short rename_scan_destination_file_2
                                <1>     ;;jb    short loc_rename_nofilename_retn
                                <1>     ;jb     loc_cmd_failed
                                <1>     ; 28/07/2022
```

```
4938 000094E7 72EF                <1>     jb      short loc_rename_failed
4939                              <1>
4940 000094E9 8935[28810100]      <1>     mov     [DestinationFilePath], esi
4941                              <1>
4942                              <1> rename_scan_destination_file_3:
4943 000094EF 46                  <1>     inc     esi
4944 000094F0 803E20              <1>     cmp     byte [esi], 20h
4945 000094F3 77FA                <1>     ja      short rename_scan_destination_file_3
4946                              <1>
4947 000094F5 C60600              <1>     mov     byte [esi], 0
4948                              <1>
4949                              <1> loc_rename_save_current_drive:
4950 000094F8 8A35[42770100]      <1>     mov     dh, [Current_Drv]
4951 000094FE 8835[9F7E0100]      <1>     mov     byte [RUN_CDRV], dh
4952                              <1>
4953                              <1> loc_rename_sf_parse_path_name:
4954 00009504 8B35[24810100]      <1>     mov     esi, [SourceFilePath]
4955 0000950A BF[E27F0100]        <1>     mov     edi, FindFile_Drv
4956 0000950F E85A140000          <1>     call    parse_path_name
4957                              <1>     ;jc     loc_cmd_failed
4958                              <1>     ; 28/07/2022
4959 00009514 72C2                <1>     jc      short loc_rename_failed
4960                              <1>
4961                              <1> loc_rename_sf_check_filename_exists:
4962 00009516 BE[24800100]        <1>     mov     esi, FindFile_Name
4963 0000951B 803E20              <1>     cmp     byte [esi], 20h
4964                              <1>     ;jna    loc_cmd_failed
4965                              <1>     ; 28/07/2022
4966 0000951E 76B8                <1>     jna     short loc_rename_failed
4967                              <1>
4968                              <1>     ;mov    [DelFile_FNPointer], esi
4969                              <1>
4970                              <1> loc_rename_sf_drv:
4971                              <1>     ;mov    dh, [Current_Drv]
4972                              <1>     ;mov    [RUN_CDRV], dh
4973                              <1>
4974 00009520 8A15[E27F0100]      <1>     mov     dl, [FindFile_Drv]
4975 00009526 38F2                <1>     cmp     dl, dh ; dh = [Current_Drv]
4976 00009528 7407                <1>     je      short rename_sf_change_directory
4977                              <1>
4978 0000952A E8B1E1FFFF          <1>     call    change_current_drive
4979                              <1>     ;jc     loc_file_rw_cmd_failed
4980                              <1>     ; 28/07/2022
4981 0000952F 722D                <1>     jc      short loc_rename_fff_failed
4982                              <1>
4983                              <1> rename_sf_change_directory:
4984 00009531 803D[E37F0100]20    <1>     cmp     byte [FindFile_Directory], 20h
4985 00009538 7614                <1>     jna     short rename_sf_find
4986                              <1>
4987 0000953A FE05[5D2E0100]      <1>     inc     byte [Restore_CDIR]
4988 00009540 BE[E37F0100]        <1>     mov     esi, FindFile_Directory
4989 00009545 30E4                <1>     xor     ah, ah ; CD_COMMAND sign -> 0
4990 00009547 E88F0E0000          <1>     call    change_current_directory
4991                              <1>     ;jc     loc_file_rw_cmd_failed
4992                              <1>     ; 28/07/2022
4993 0000954C 7210                <1>     jc      short loc_rename_fff_failed
4994                              <1>
4995                              <1> ;rename_sf_change_prompt_dir_string:
4996                              <1>     ;call   change_prompt_dir_string
4997                              <1>
4998                              <1> rename_sf_find:
4999                              <1>     ;mov    esi, [DelFile_FNPointer]
5000 0000954E BE[24800100]        <1>     mov     esi, FindFile_Name
5001                              <1>
5002 00009553 66B80008            <1>     mov     ax, 0800h ; Except volume labels
5003 00009557 E80EF4FFFF          <1>     call    find_first_file
5004                              <1>     ;jc     loc_file_rw_cmd_failed
5005                              <1>     ; 28/07/2022
5006 0000955C 7305                <1>     jnc     short loc_rename_sf_ambgfn_check
5007                              <1>
5008                              <1> loc_rename_fff_failed:
5009 0000955E E9D5F7FFFF          <1>     jmp     loc_file_rw_cmd_failed
5010                              <1>
5011                              <1> loc_rename_sf_ambgfn_check:
5012 00009563 6621D2              <1>     and     dx, dx ; Ambiguous filename chars used sign (DX>0)
5013                              <1>     ;       (Note: It was BX in TRDOS v1)
5014                              <1>     ;;jz    short loc_rename_sf_found
5015                              <1>     ;jnz    loc_file_not_found
5016                              <1>     ; 28/07/2022
5017 00009566 7405                <1>     jz      short loc_rename_sf_found
5018 00009568 E9CFCFFFF           <1>     jmp     loc_file_not_found
5019                              <1>
5020                              <1>     ;mov    eax, 2 ; File not found sign
5021                              <1>     ;stc
5022                              <1>     ;jmp    loc_file_rw_cmd_failed
5023                              <1>
5024                              <1> loc_rename_sf_found:
5025                              <1>     ; EDI = Directory buffer entry offset/address
5026                              <1>     ; BL = File (or Directory) Attributes
5027                              <1>     ;       (Note: It was 'CL' in TRDOS v1)
5028                              <1>     ; mov  bl, [EDI+0Bh]
5029                              <1>
5030 0000956D F6C307              <1>     test    bl, 07h ; Attributes, S-H-R
5031                              <1>     ;jnz    loc_permission_denied
5032                              <1>     ; 28/07/2022
5033 00009570 7405                <1>     jz      short loc_rename_attrb_ok
5034 00009572 E9CBF7FFFF          <1>     jmp     loc_permission_denied
5035                              <1>
5036                              <1> loc_rename_attrb_ok:
5037 00009577 BE[E27F0100]        <1>         mov     esi, FindFile_Drv
5038 0000957C BF[2C810100]        <1>         mov     edi, SourceFile_Drv
5039 00009581 B920000000          <1>     mov     ecx, 32
5040 00009586 F3A5                <1>     rep     movsd
5041                              <1>
5042                              <1> loc_rename_df_parse_path_name:
5043 00009588 8B35[28810100]      <1>     mov     esi, [DestinationFilePath]
5044 0000958E BF[E27F0100]        <1>     mov     edi, FindFile_Drv
5045 00009593 E8D6130000          <1>     call    parse_path_name
5046 00009598 7219                <1>     jc      short loc_rename_df_cmd_failed
5047                              <1>
5048                              <1>     ;mov    dh, [RUN_CDRV]
5049 0000959A 8A35[42770100]      <1>     mov     dh, [Current_Drv]
5050                              <1>
5051                              <1>     ; 'rename' command is valid only for same dos drive and same dir!
5052                              <1>     ; ('move' command must be used if source file and destination file
5053                              <1>     ; directories are not same!)
5054 000095A0 8A15[E27F0100]      <1>     mov     dl, [FindFile_Drv]
5055 000095A6 38F2                <1>     cmp     dl, dh ; are source and destination drives different ?!
5056 000095A8 7509                <1>     jne     short loc_rename_df_cmd_failed ; yes!
5057                              <1>
5058                              <1> rename_df_check_dirname_exists:
5059 000095AA 803D[E37F0100]00    <1>     cmp     byte [FindFile_Directory], 0
5060 000095B1 760B                <1>     jna     short rename_df_check_filename_exists
5061                              <1>
```

```
                                <1>     ; different source file and destination file directories !
                                <1> loc_rename_df_cmd_failed:
000095B3 B801000000             <1>     mov     eax, 1 ; TRDOS 'Bad command or file name' error
000095B8 F9                     <1>     stc
000095B9 E97AF7FFFF             <1>     jmp     loc_file_rw_cmd_failed
                                <1>
                                <1> rename_df_check_filename_exists:
000095BE BE[24800100]           <1>     mov     esi, FindFile_Name
000095C3 E801F7FFFF             <1>     call    check_filename
                                <1>     ;jc     loc_mkdir_invalid_dir_name_chars
                                <1>     ; 28/07/2022
000095C8 7305                   <1>     jnc     short loc_rename_file_name_ok
000095CA E91BF8FFFF             <1>     jmp     loc_mkdir_invalid_dir_name_chars
                                <1>
                                <1> loc_rename_file_name_ok:
                                <1>     ;mov    [DelFile_FNPointer], esi
                                <1>     ;cmp    byte [esi], 20h
                                <1>     ;ja     short loc_rename_df_find
                                <1>
                                <1>     ;mov    dh, [Current_Drv] ; dh has not been changed
                                <1>
                                <1> rename_df_drv_check_writable:
                                <1>     ;movzx  esi, dh
                                <1>     ;;movzx esi, byte [Current_Drv]
                                <1>     ;add    esi, Logical_DOSDisks
                                <1>     ; 25/07/2025 (BugFix)
000095CF 31C0                   <1>     xor     eax, eax
000095D1 88F4                   <1>     mov     ah, dh
000095D3 BE00010900             <1>     mov     esi, Logical_DOSDisks
000095D8 01C6                   <1>     add     esi, eax
                                <1>
000095DA 88F2                   <1>     mov     dl, dh ; dl = [Current_Drv]
000095DC 8A7601                 <1>     mov     dh, [esi+LD_DiskType]
                                <1>
000095DF 80FE01                 <1>     cmp     dh, 1 ; 0 = Invalid
000095E2 7310                   <1>     jnb     short rename_df_compare_sf_df_name
                                <1>
                                <1>     ; 16/10/2016 (13h -> 30)
000095E4 B81E000000             <1>     mov     eax, 30 ; 'Disk write-protected' error
000095E9 8B1D[28810100]         <1>     mov     ebx, [DestinationFilePath]
000095EF E944F7FFFF             <1>     jmp     loc_file_rw_cmd_failed
                                <1>
                                <1> rename_df_compare_sf_df_name:
000095F4 BE[24800100]           <1>     mov     esi, FindFile_Name
000095F9 BF[6E810100]           <1>     mov     edi, SourceFile_Name
                                <1>     ;mov    ecx, 12
                                <1>     ; 28/07/2022
000095FE 29C9                   <1>     sub     ecx, ecx
00009600 B10C                   <1>     mov     cl, 12
                                <1> rename_df_compare_sf_df_name_next:
00009602 AC                     <1>     lodsb
00009603 AE                     <1>     scasb
00009604 7506                   <1>     jne     short loc_rename_df_find
00009606 08C0                   <1>     or      al, al
00009608 74A9                   <1>     jz      short loc_rename_df_cmd_failed
0000960A E2F6                   <1>     loop    rename_df_compare_sf_df_name_next
                                <1>
                                <1> loc_rename_df_find:
                                <1>     ;mov    esi, [DelFile_FNPointer]
0000960C BE[24800100]           <1>     mov     esi, FindFile_Name
                                <1>
                                <1>     ;xor    ax, ax ; Any
                                <1>     ; 28/07/2022
00009611 31C0                   <1>     xor     eax, eax ; 0 ; Any
00009613 E852F3FFFF             <1>     call    find_first_file
                                <1>     ;;jnc   short loc_rename_df_found
                                <1>     ;; 29/12/2017
                                <1>     ;jnc    loc_permission_denied
                                <1>     ; 28/07/2022
00009618 7205                   <1>     jc      short loc_rename_df_check_error_code
0000961A E923F7FFFF             <1>     jmp     loc_permission_denied
                                <1>
                                <1> loc_rename_df_check_error_code:
                                <1>     ;cmp    eax, 2
0000961F 3C02                   <1>     cmp     al, 2 ; Not found error
00009621 7406                   <1>     je      short rename_df_move_find_struct_to_dest
00009623 F9                     <1>     stc
00009624 E90FF7FFFF             <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:
00009629 BE[E27F0100]           <1>         mov     esi, FindFile_Drv
0000962E BF[AC810100]           <1>         mov     edi, DestinationFile_Drv
                                <1>     ;mov    ecx, 32
                                <1>     ; 28/07/2022
00009633 29C9                   <1>     sub     ecx, ecx
00009635 B120                   <1>     mov     cl, 32
00009637 F3A5                   <1>     rep     movsd
                                <1>
                                <1> loc_rename_df_process_q_sf:
                                <1>     ;mov    ecx, 12
00009639 B10C                   <1>     mov     cl, 12
0000963B BE[6E810100]           <1>     mov     esi, SourceFile_Name
00009640 BF[B5330100]           <1>     mov     edi, Rename_OldName
                                <1> rename_df_process_q_nml_1_sf:
00009645 AC                     <1>     lodsb
00009646 3C20                   <1>         cmp al, 20h
00009648 7603                   <1>         jna short rename_df_process_q_nml_2_sf
0000964A AA                     <1>     stosb
0000964B E2F8                   <1>     loop    rename_df_process_q_nml_1_sf
                                <1>
                                <1> rename_df_process_q_nml_2_sf:
0000964D C60700                 <1>     mov     byte [edi], 0
                                <1>
                                <1> loc_rename_df_process_q_df:
                                <1>     ;mov    ecx, 12
00009650 B10C                   <1>     mov     cl, 12
00009652 BE[EE810100]           <1>     mov     esi, DestinationFile_Name
00009657 BF[C6330100]           <1>     mov     edi, Rename_NewName
                                <1> rename_df_process_q_nml_1_df:
0000965C AC                     <1>     lodsb
0000965D 3C20                   <1>     cmp     al, 20h
0000965F 7603                   <1>     jna     short loc_rename_df_process_q_nml_2_df
00009661 AA                     <1>     stosb
00009662 E2F8                   <1>     loop    rename_df_process_q_nml_1_df
                                <1>
                                <1> loc_rename_df_process_q_nml_2_df:
00009664 C60700                 <1>     mov     byte [edi], 0
```

```
5186                                    <1>
5187                                    <1> loc_rename_confirmation_question:
5188 00009667 BE[8D330100]             <1>     mov     esi, Msg_DoYouWantRename
5189 0000966C E839D7FFFF               <1>     call    print_msg
5190                                    <1>
5191 00009671 A0[89810100]             <1>     mov     al, [SourceFile_DirEntry+11] ; Attributes
5192 00009676 2410                     <1>     and     al, 10h
5193 00009678 750C                     <1>     jnz     short rename_confirmation_question_dir
5194                                    <1>
5195                                    <1> rename_confirmation_question_file:
5196 0000967A BE[A4330100]             <1>     mov     esi, Rename_File
5197 0000967F E826D7FFFF               <1>     call    print_msg
5198 00009684 EB0A                     <1>     jmp     short rename_confirmation_question_as
5199                                    <1>
5200                                    <1> rename_confirmation_question_dir:
5201 00009686 BE[AA330100]             <1>     mov     esi, Rename_Directory
5202 0000968B E81AD7FFFF               <1>     call    print_msg
5203                                    <1>
5204                                    <1> rename_confirmation_question_as:
5205 00009690 BE[B5330100]             <1>     mov     esi, Rename_OldName
5206 00009695 E810D7FFFF               <1>     call    print_msg
5207 0000969A BE[C2330100]             <1>     mov     esi, Msg_File_rename_as
5208 0000969F E806D7FFFF               <1>     call    print_msg
5209 000096A4 BE[E9320100]             <1>     mov     esi, Msg_YesNo
5210 000096A9 E8FCD6FFFF               <1>     call    print_msg
5211                                    <1>
5212                                    <1> loc_rename_ask_again:
5213 000096AE 30E4                     <1>     xor     ah, ah
5214 000096B0 E86778FFFF               <1>     call    int16h
5215 000096B5 3C1B                     <1>     cmp     al, 1Bh
5216 000096B7 740F                     <1>     je      short loc_do_not_rename_file
5217 000096B9 24DF                     <1>     and     al, 0DFh
5218 000096BB A2[F3320100]             <1>     mov     [Y_N_nextline], al
5219 000096C0 3C59                     <1>     cmp     al, 'Y'
5220 000096C2 7404                     <1>     je      short loc_yes_rename_file
5221 000096C4 3C4E                     <1>     cmp     al, 'N'
5222 000096C6 75E6                     <1>     jne     short loc_rename_ask_again
5223                                    <1>
5224                                    <1> loc_do_not_rename_file:
5225                                    <1> loc_yes_rename_file:
5226 000096C8 E89EF7FFFF               <1>     call    y_n_answer ; 29/12/2017
5227                                    <1>     ;cmp    al, 'Y' ; 'yes'
5228                                    <1>     ;cmc
5229                                    <1>         ;jnc       loc_file_rw_restore_retn
5230 000096CD 3C4E                     <1>     cmp     al, 'N' ; 'no'
5231                                    <1>     ;je     loc_file_rw_restore_retn
5232                                    <1>     ; 28/07/2022
5233 000096CF 7505                     <1>     jne     short loc_rename_file_yes
5234 000096D1 E962F6FFFF               <1>     jmp     loc_file_rw_restore_retn
5235                                    <1>
5236                                    <1> loc_rename_file_yes: ; 28/07/2022
5237 000096D6 BE[C6330100]             <1>     mov     esi, Rename_NewName
5238 000096DB 668B0D[A6810100]         <1>     mov     cx, [SourceFile_DirEntryNumber]
5239 000096E2 66A1[92810100]           <1>     mov     ax, [SourceFile_DirEntry+20] ; First Cluster, HW
5240 000096E8 C1E010                   <1>     shl     eax, 16 ; 13/11/2017
5241 000096EB 66A1[98810100]           <1>     mov     ax, [SourceFile_DirEntry+26] ; First Cluster, LW
5242                                    <1>
5243 000096F1 0FB61D[7B810100]         <1>     movzx   ebx, byte [SourceFile_LongNameEntryLength]
5244 000096F8 E8201A0000               <1>     call    rename_directory_entry
5245 000096FD E94FF7FFFF               <1>     jmp     loc_rename_file_ok
5246                                    <1> ;loc_rename_file_ok:
5247                                    <1> ;   jc      loc_run_cmd_failed
5248                                    <1> ;   mov     esi, Msg_OK
5249                                    <1> ;   call    proc_printmsg
5250                                    <1> ;   jmp     loc_file_rw_restore_retn
5251                                    <1>
5252                                    <1> move_file:
5253                                    <1>     ; 07/08/2022
5254                                    <1>     ; 28/07/2022 (TRDOS 386 Kernel v2.0.5)
5255                                    <1>     ; 11/03/2016
5256                                    <1>     ; 09/03/2016
5257                                    <1>     ; 08/03/2016 (TRDOS 386 = TRDOS v2.0)
5258                                    <1>     ; 21/05/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_move')
5259                                    <1>     ; 23/04/2011
5260                                    <1>
5261                                    <1> get_move_source_fchar:
5262                                    <1>     ; esi = file name
5263 00009702 803E20                   <1>     cmp     byte [esi], 20h
5264 00009705 7613                     <1>         jna short loc_move_nofilename_retn
5265                                    <1>
5266 00009707 8935[24810100]           <1>     mov     [SourceFilePath], esi
5267                                    <1>
5268                                    <1> move_scan_source_file:
5269 0000970D 46                       <1>     inc     esi
5270 0000970E 803E20                   <1>     cmp     byte [esi], 20h
5271 00009711 7408                     <1>         je      short move_scan_destination_1
5272                                    <1>     ;;jb    short loc_move_nofilename_retn
5273                                    <1>     ;jb     loc_cmd_failed
5274                                    <1>     ;jmp    short move_scan_source_file
5275                                    <1>     ; 28/07/2022
5276 00009713 77F8                     <1>     ja      short move_scan_source_file
5277                                    <1> loc_move_failed:
5278 00009715 E92AEDFFFF               <1>     jmp     loc_cmd_failed
5279                                    <1>
5280                                    <1> loc_move_nofilename_retn:
5281 0000971A C3                       <1>     retn
5282                                    <1>
5283                                    <1> move_scan_destination_1:
5284 0000971B C60600                   <1>     mov     byte [esi], 0
5285                                    <1>
5286                                    <1> move_scan_destination_2:
5287 0000971E 46                       <1>     inc     esi
5288 0000971F 803E20                   <1>     cmp     byte [esi], 20h
5289 00009722 74FA                     <1>     je      short move_scan_destination_2
5290                                    <1>     ;;jb    short loc_move_nofilename_retn
5291                                    <1>     ;jb     loc_cmd_failed
5292                                    <1>     ; 28/07/2022
5293 00009724 72EF                     <1>     jb      short loc_move_failed
5294                                    <1>
5295 00009726 8935[28810100]           <1>     mov     [DestinationFilePath], esi
5296                                    <1>
5297                                    <1> move_scan_destination_3:
5298 0000972C 46                       <1>     inc     esi
5299 0000972D 803E20                   <1>     cmp     byte [esi], 20h
5300 00009730 77FA                     <1>     ja      short move_scan_destination_3
5301 00009732 C60600                   <1>     mov     byte [esi], 0
5302                                    <1>
5303                                    <1> loc_move_scan_destination_OK:
5304 00009735 8B35[24810100]           <1>     mov     esi, [SourceFilePath]
5305 0000973B 8B3D[28810100]           <1>     mov     edi, [DestinationFilePath]
5306                                    <1>
5307 00009741 B001                     <1>     mov     al, 1  ; move procedure Phase 1
5308 00009743 E8411A0000               <1>     call    move_source_file_to_destination_file
5309 00009748 7325                     <1>     jnc     short move_source_file_to_destination_question
```

```
5310                                    <1>
5311                                    <1> loc_move_cmd_failed_1:
5312 0000974A 08C0                      <1>     or      al, al
5313                                    <1>     ;jz     loc_cmd_failed
5314                                    <1>     ; 28/07/2022
5315 0000974C 74C7                      <1>     jz      short loc_move_failed
5316                                    <1>
5317 0000974E 3C11                      <1>     cmp     al, 11h
5318 00009750 7409                      <1>     je      short loc_msg_not_same_device
5319                                    <1>     ;cmp    al, 05h
5320                                    <1>     ;cmp    al, ERR_PERM_DENIED ; 29/12/2017
5321                                    <1>     ;jne    loc_run_cmd_failed
5322                                    <1>     ;jmp    loc_permission_denied
5323 00009752 3C0B                      <1>     cmp     al, ERR_PERM_DENIED
5324                                    <1>     ;je     loc_permission_denied
5325                                    <1>     ; 28/07/2022
5326 00009754 7414                      <1>     je      short loc_move_perm_denied
5327 00009756 E914EDFFFF                <1>     jmp     loc_run_cmd_failed
5328                                    <1>
5329                                    <1>     ;mov    esi, Msg_Permission_denied
5330                                    <1>     ;call   print_msg
5331                                    <1>     ;jmp    loc_file_rw_restore_retn
5332                                    <1>
5333                                    <1> loc_msg_not_same_device:
5334 0000975B BE[D3330100]              <1>     mov     esi, msg_not_same_drv
5335 00009760 E845D6FFFF                <1>     call    print_msg
5336 00009765 E9CEF5FFFF                <1>     jmp     loc_file_rw_restore_retn
5337                                    <1>
5338                                    <1>     ; 28/07/2022
5339                                    <1> loc_move_perm_denied:
5340 0000976A E9D3F5FFFF                <1>     jmp     loc_permission_denied
5341                                    <1>
5342                                    <1> move_source_file_to_destination_question:
5343 0000976F A0[2C810100]              <1>         mov     al, [SourceFile_Drv]
5344 00009774 0441                      <1>     add     al, 'A'
5345 00009776 A2[35340100]              <1>     mov     [msg_source_file_drv], al
5346 0000977B A0[AC810100]              <1>         mov     al, [DestinationFile_Drv]
5347 00009780 0441                      <1>     add     al, 'A'
5348 00009782 A2[54340100]              <1>     mov     [msg_destination_file_drv], al
5349                                    <1>
5350 00009787 57                        <1>     push    edi ; *
5351                                    <1>
5352 00009788 BE[19340100]              <1>     mov     esi, msg_source_file
5353 0000978D E818D6FFFF                <1>     call    print_msg
5354 00009792 BE[2D810100]              <1>     mov     esi, SourceFile_Directory
5355 00009797 803E20                    <1>     cmp     byte [esi], 20h
5356 0000979A 7605                      <1>     jna     short msftdfq_sfn
5357 0000979C E809D6FFFF                <1>     call    print_msg
5358                                    <1> msftdfq_sfn:
5359 000097A1 BE[6E810100]              <1>     mov     esi, SourceFile_Name
5360 000097A6 E8FFD5FFFF                <1>     call    print_msg
5361 000097AB BE[38340100]              <1>     mov     esi, msg_destination_file
5362 000097B0 E8F5D5FFFF                <1>     call    print_msg
5363 000097B5 BE[AD810100]              <1>     mov     esi, DestinationFile_Directory
5364 000097BA 803E20                    <1>     cmp     byte [esi], 20h
5365 000097BD 7605                      <1>     jna     short msftdfq_dfn
5366 000097BF E8E6D5FFFF                <1>     call    print_msg
5367                                    <1> msftdfq_dfn:
5368 000097C4 BE[EE810100]              <1>     mov     esi, DestinationFile_Name
5369 000097C9 E8DCD5FFFF                <1>     call    print_msg
5370 000097CE BE[57340100]              <1>     mov     esi, msg_copy_nextline
5371 000097D3 E8D2D5FFFF                <1>     call    print_msg
5372 000097D8 BE[57340100]              <1>     mov     esi, msg_copy_nextline
5373 000097DD E8C8D5FFFF                <1>     call    print_msg
5374                                    <1>
5375                                    <1> loc_move_ask_for_new_file_yes_no:
5376 000097E2 BE[E5330100]              <1>     mov     esi, Msg_DoYouWantMoveFile
5377 000097E7 E8BED5FFFF                <1>     call    print_msg
5378 000097EC BE[E9320100]              <1>     mov     esi, Msg_YesNo
5379 000097F1 E8B4D5FFFF                <1>     call    print_msg
5380                                    <1> loc_move_ask_for_new_file_again:
5381 000097F6 30E4                      <1>     xor     ah, ah
5382 000097F8 E81F77FFFF                <1>     call    int16h
5383 000097FD 3C1B                      <1>     cmp     al, 1Bh
5384                                    <1>     ;je     short loc_do_not_move_file
5385 000097FF 743F                      <1>     je      short loc_move_y_n_escape
5386 00009801 24DF                      <1>     and     al, 0DFh
5387 00009803 A2[F3320100]              <1>         mov     [Y_N_nextline], al
5388 00009808 3C59                      <1>     cmp     al, 'Y'
5389 0000980A 7404                      <1>     je      short loc_yes_move_file
5390 0000980C 3C4E                      <1>     cmp     al, 'N'
5391 0000980E 75E6                      <1>     jne     short loc_move_ask_for_new_file_again
5392                                    <1>
5393                                    <1> loc_do_not_move_file:
5394                                    <1> loc_yes_move_file:
5395 00009810 E856F6FFFF                <1>     call    y_n_answer ; 29/12/2017
5396 00009815 5F                        <1>     pop     edi ; *
5397                                    <1>     ;cmp    al, 'Y' ; 'yes'
5398                                    <1>     ;cmc
5399                                    <1>         ;jnc       loc_file_rw_restore_retn
5400 00009816 3C4E                      <1>     cmp     al, 'N' ; 'no'
5401                                    <1>         ;je loc_file_rw_restore_retn
5402                                    <1>     ; 28/07/2022
5403 00009818 7421                      <1>     je      short loc_move_rw_restore_retn
5404                                    <1>
5405                                    <1> loc_move_yes_move_file:
5406 0000981A B002                      <1>     mov     al, 2 ; move procedure Phase 2
5407 0000981C E868190000                <1>     call    move_source_file_to_destination_file
5408                                    <1>     ;;jc    short loc_move_cmd_failed_2
5409                                    <1>         ;jnc       move_source_file_to_dest_OK
5410                                    <1>     ; 28/07/2022
5411 00009821 7205                      <1>     jc      short loc_move_cmd_failed_2
5412                                    <1>     ; 07/08/2022
5413 00009823 E930F6FFFF                <1>     jmp     move_source_file_to_dest_OK
5414                                    <1>
5415                                    <1> ;move_source_file_to_destination_OK:
5416                                    <1> ;   mov     esi, Msg_OK
5417                                    <1> ;   call    print_msg
5418                                    <1> ;   jmp     loc_file_rw_restore_retn
5419                                    <1>
5420                                    <1> loc_move_cmd_failed_2:
5421 00009828 3C27                      <1>     cmp     al, 27h
5422                                    <1>     ;jne    loc_run_cmd_failed
5423                                    <1>     ; 28/07/2022
5424 0000982A 7405                      <1>     je      short loc_move_ids_err
5425 0000982C E93EECFFFF                <1>     jmp     loc_run_cmd_failed
5426                                    <1>
5427                                    <1> loc_move_ids_err: ; 28/07/2022
5428 00009831 BE[FE330100]              <1>     mov     esi, msg_insufficient_disk_space
5429 00009836 E86FD5FFFF                <1>     call    print_msg
5430                                    <1>
5431                                    <1> loc_move_rw_restore_retn: ; 28/07/2022
5432 0000983B E9F8F4FFFF                <1>     jmp     loc_file_rw_restore_retn
5433                                    <1>
```

```
                                <1> loc_move_y_n_escape:
00009840 B04E                   <1>     mov     al, 'N' ; 'no'
00009842 EBCC                   <1>     jmp     short loc_do_not_move_file
                                <1>
                                <1> copy_file:
                                <1>     ; 31/08/2024 - TRDOS 386 v2.0.9
                                <1>     ; 25/07/2022 - TRDOS 386 Kernel v2.0.5
                                <1>     ; 15/10/2016
                                <1>     ; 24/03/2016
                                <1>     ; 21/03/2016
                                <1>     ; 15/03/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>     ; 21/05/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_copy')
                                <1>     ; 01/08/2010
                                <1>
                                <1> get_copy_source_fchar:
                                <1>     ; esi = file name
00009844 803E20                 <1>     cmp     byte [esi], 20h
00009847 7613                   <1>         jna     short loc_copy_nofilename_retn
                                <1>
00009849 8935[24810100]         <1>     mov     [SourceFilePath], esi
                                <1>
                                <1> copy_scan_source_file:
0000984F 46                     <1>     inc     esi
00009850 803E20                 <1>     cmp     byte [esi], 20h
00009853 7408                   <1>     je      short copy_scan_destination_1
                                <1>     ;;jb    short loc_copy_nofilename_retn
                                <1>     ;jb     loc_cmd_failed
                                <1>     ;jmp    short copy_scan_source_file
                                <1>     ; 25/07/2022
00009855 73F8                   <1>     jnb     short copy_scan_source_file
                                <1> copy_scan_destination_0:
00009857 E9E8EBFFFF             <1>     jmp     loc_cmd_failed
                                <1>
                                <1> loc_copy_nofilename_retn:
0000985C C3                     <1>     retn
                                <1>
                                <1> copy_scan_destination_1:
0000985D C60600                 <1>     mov     byte [esi], 0
                                <1>
                                <1> copy_scan_destination_2:
00009860 46                     <1>     inc     esi
00009861 803E20                 <1>     cmp     byte [esi], 20h
00009864 74FA                   <1>     je      short copy_scan_destination_2
                                <1>     ;;jb    short loc_copy_nofilename_retn
                                <1>     ;jb     loc_cmd_failed
                                <1>     ; 25/07/2022
00009866 72EF                   <1>     jb      short copy_scan_destination_0
                                <1>
00009868 8935[28810100]         <1>     mov     [DestinationFilePath], esi
                                <1>
                                <1> copy_scan_destination_3:
0000986E 46                     <1>     inc     esi
0000986F 803E20                 <1>     cmp     byte [esi], 20h
00009872 77FA                   <1>     ja      short copy_scan_destination_3
00009874 C60600                 <1>     mov     byte [esi], 0
                                <1>
                                <1> loc_copy_save_current_drive:
00009877 8A35[42770100]         <1>     mov     dh, [Current_Drv]
0000987D 8835[9F7E0100]         <1>     mov     [RUN_CDRV], dh
                                <1>
                                <1> copy_source_file_to_destination_phase_1:
00009883 8B35[24810100]         <1>     mov     esi, [SourceFilePath]
00009889 8B3D[28810100]         <1>     mov     edi, [DestinationFilePath]
                                <1>
0000988F B001                   <1>     mov     al, 1  ; copy procedure Phase 1
00009891 E8571B0000             <1>     call    copy_source_file_to_destination_file
00009896 7327                   <1>     jnc     short copy_source_file_to_destination_question
                                <1>
                                <1> loc_copy_cmd_failed_1:
                                <1>     ; 18/03/2016 (restore current drive and directory)
00009898 08C0                   <1>     or      al, al
0000989A 7507                   <1>     jnz     short loc_copy_cmd_failed_2
                                <1>
0000989C FEC0                   <1>         inc     al ; mov al, 1 ; Bad command or file name !
                                <1> loc_copy_cmd_failed_3:     ; 25/07/2022
0000989E E9CCEBFFFF             <1>     jmp     loc_run_cmd_failed
                                <1>
                                <1> loc_copy_cmd_failed_2:
000098A3 3C27                   <1>     cmp     al, 27h ; Insufficient disk space
000098A5 7409                   <1>     je      short loc_file_write_insuff_disk_space_msg
                                <1>
                                <1>     ; 29/12/2017
                                <1>     ;cmp    al, 05h
000098A7 3C0B                   <1>     cmp     al, ERR_PERM_DENIED
                                <1>     ;jne    loc_run_cmd_failed
                                <1>     ; 25/07/2022
000098A9 75F3                   <1>     jne     short loc_copy_cmd_failed_3
                                <1>
000098AB E992F4FFFF             <1>     jmp     loc_permission_denied
                                <1>
                                <1> loc_file_write_insuff_disk_space_msg:
000098B0 BE[FE330100]           <1>     mov     esi, msg_insufficient_disk_space
000098B5 E8F0D4FFFF             <1>     call    print_msg
000098BA E979F4FFFF             <1>         jmp     loc_file_rw_restore_retn
                                <1>
                                <1> copy_source_file_to_destination_question:
000098BF 57                     <1>     push    edi ; *
                                <1>
                                <1>     ; dh = source file attributes
                                <1>     ; dl > 0 -> destination file found
000098C0 20D2                   <1>     and     dl, dl
000098C2 7446                   <1>     jz      short copy_source_file_to_destination_pass_owrq
                                <1>
                                <1> loc_copy_ask_for_owr_yes_no:
000098C4 BE[5A340100]           <1>     mov     esi, Msg_DoYouWantOverWriteFile
000098C9 E8DCD4FFFF             <1>     call    print_msg
000098CE BE[EE810100]           <1>     mov     esi, DestinationFile_Name
000098D3 E8D2D4FFFF             <1>     call    print_msg
000098D8 BE[E9320100]           <1>     mov     esi, Msg_YesNo
000098DD E8C8D4FFFF             <1>     call    print_msg
                                <1>
                                <1> loc_copy_ask_for_owr_again:
000098E2 30E4                   <1>     xor     ah, ah
000098E4 E83376FFFF             <1>     call    int16h
000098E9 3C1B                   <1>     cmp     al, 1Bh
                                <1>         ;je     loc_do_not_copy_file
000098EB 7419                   <1>         je      short loc_copy_y_n_escape
000098ED 24DF                   <1>     and     al, 0DFh
000098EF A2[F3320100]           <1>         mov     [Y_N_nextline], al
000098F4 3C59                   <1>     cmp     al, 'Y'
                                <1>     ;je     loc_yes_copy_file
                                <1>     ; 25/07/2022
000098F6 7505                   <1>     jne     short loc_copy_ask_for_owr_n
000098F8 E9AD000000             <1>     jmp     loc_yes_copy_file
```

```
                                <1>
                                <1> loc_copy_ask_for_owr_n: ; 25/07/2022
000098FD 3C4E                   <1>     cmp     al, 'N'
                                <1>         ;je loc_do_not_copy_file
                                <1>     ;jmp    short loc_copy_ask_for_owr_again
                                <1>     ; 25/07/2022
000098FF 75E1                   <1>     jne     short loc_copy_ask_for_owr_again
                                <1> loc_do_not_copy_file_j:
00009901 E9A4000000             <1>     jmp     loc_do_not_copy_file
                                <1>
                                <1> loc_copy_y_n_escape:
00009906 B04E                   <1>     mov     al, 'N' ; 'no'
                                <1>     ;jmp    loc_do_not_copy_file
                                <1>     ; 25/07/2022
00009908 EBF7                   <1>     jmp     short loc_do_not_copy_file_j
                                <1>
                                <1> copy_source_file_to_destination_pass_owrq:
0000990A A0[2C810100]           <1>     mov      al, [SourceFile_Drv]
0000990F 0441                   <1>     add     al, 'A'
00009911 A2[35340100]           <1>     mov     [msg_source_file_drv], al
00009916 A0[AC810100]           <1>         mov     al, [DestinationFile_Drv]
0000991B 0441                   <1>     add     al, 'A'
0000991D A2[54340100]           <1>     mov     [msg_destination_file_drv], al
                                <1>
00009922 BE[19340100]           <1>     mov     esi, msg_source_file
00009927 E87ED4FFFF             <1>     call    print_msg
0000992C BE[2D810100]           <1>     mov     esi, SourceFile_Directory
00009931 803E20                 <1>     cmp     byte [esi], 20h
00009934 7605                   <1>     jna     short csftdfq_sfn
00009936 E86FD4FFFF             <1>     call    print_msg
                                <1> csftdfq_sfn:
0000993B BE[6E810100]           <1>     mov     esi, SourceFile_Name
00009940 E865D4FFFF             <1>     call    print_msg
00009945 BE[38340100]           <1>     mov     esi, msg_destination_file
0000994A E85BD4FFFF             <1>     call    print_msg
0000994F BE[AD810100]           <1>     mov     esi, DestinationFile_Directory
00009954 803E20                 <1>     cmp     byte [esi], 20h
00009957 7605                   <1>     jna     short csftdfq_dfn
00009959 E84CD4FFFF             <1>     call    print_msg
                                <1> csftdfq_dfn:
0000995E BE[EE810100]           <1>     mov     esi, DestinationFile_Name
00009963 E842D4FFFF             <1>     call    print_msg
00009968 BE[57340100]           <1>     mov     esi, msg_copy_nextline
0000996D E838D4FFFF             <1>     call    print_msg
00009972 BE[57340100]           <1>     mov     esi, msg_copy_nextline
00009977 E82ED4FFFF             <1>     call    print_msg
                                <1>
                                <1> loc_copy_ask_for_new_file_yes_no:
0000997C BE[79340100]           <1>     mov     esi, Msg_DoYouWantCopyFile
00009981 E824D4FFFF             <1>     call    print_msg
00009986 BE[E9320100]           <1>     mov     esi, Msg_YesNo
0000998B E81AD4FFFF             <1>     call    print_msg
                                <1>
                                <1> loc_copy_ask_for_new_file_again:
00009990 30E4                   <1>     xor     ah, ah
00009992 E88575FFFF             <1>     call    int16h
00009997 3C1B                   <1>     cmp     al, 1Bh
00009999 740F                   <1>     je      short loc_do_not_copy_file
0000999B 24DF                   <1>     and     al, 0DFh
0000999D A2[F3320100]           <1>         mov     [Y_N_nextline], al
000099A2 3C59                   <1>     cmp     al, 'Y'
000099A4 7404                   <1>     je      short loc_yes_copy_file
000099A6 3C4E                   <1>     cmp     al, 'N'
000099A8 75E6                   <1>     jne     short loc_copy_ask_for_new_file_again
                                <1>
                                <1> loc_do_not_copy_file:
                                <1> loc_yes_copy_file:
000099AA E8BCF4FFFF             <1>     call    y_n_answer ; 29/12/2017
000099AF 5F                     <1>     pop     edi ; *
                                <1>     ;cmp    al, 'Y' ; 'yes'
                                <1>     ;cmc
                                <1>         ;jnc        loc_file_rw_restore_retn
000099B0 3C4E                   <1>     cmp     al, 'N' ; 'no'
                                <1>     ;je     loc_file_rw_restore_retn
                                <1>     ; 25/07/2022
000099B2 7505                   <1>     jne     short copy_source_file_to_destination_pass_q
000099B4 E97FF3FFFF             <1>     jmp     loc_file_rw_restore_retn
                                <1>
                                <1> copy_source_file_to_destination_pass_q:
000099B9 B002                   <1>     mov     al, 2  ; copy procedure Phase 2
000099BB E82D1A0000             <1>     call    copy_source_file_to_destination_file
                                <1>     ;jc     short loc_file_write_check_disk_space_err
                                <1>
                                <1>     ; 31/08/2024
000099C0 9C                     <1>     pushf
                                <1>
                                <1>     ; 24/03/2016
                                <1>     ;;push  cx
                                <1>     ;push   ecx ; 29/12/2017
                                <1>
000099C1 BE[57340100]           <1>     mov     esi, msg_copy_nextline
000099C6 E8DFD3FFFF             <1>     call    print_msg
                                <1>
                                <1>     ;pop    eax ; 29/12/2017
                                <1>     ;;pop   cx
                                <1>     ;pop    ax
                                <1>
                                <1>     ; 31/08/2024
000099CB 9D                     <1>     popf
000099CC 7305                   <1>     jnc     short copy_source_file_to_destination_OK
                                <1>
                                <1>     ; 31/08/2024
                                <1>     ;;or    cl, cl
                                <1>     ;;or    al, al
                                <1>     ;jz     short copy_source_file_to_destination_OK
                                <1>     ;
                                <1>     ;; 15/10/2016 (1Dh -> 18)
                                <1>     ;; 18/03/2016 (1Dh)
                                <1>     ;;cmp   cl, 18 ; write error
                                <1>     ;cmp    al, 18
                                <1>     ;jne    short copy_source_file_to_destination_not_OK
                                <1>     ;;
                                <1>     ;;mov   al, cl ; error number (write fault!)
                                <1>     ;stc
                                <1>
000099CE E965F3FFFF             <1>     jmp     loc_file_rw_cmd_failed
                                <1>
                                <1>     ; 31/08/2024
                                <1> ;copy_source_file_to_destination_not_OK:
                                <1>     ;mov    esi, Msg_read_file_error_before_EOF
                                <1>     ;call   print_msg
                                <1>     ;jmp    loc_file_rw_restore_retn
                                <1>
                                <1> copy_source_file_to_destination_OK:
```

```
5682 000099D3 BE[F7320100]        <1>     mov     esi, Msg_OK
5683 000099D8 E8CDD3FFFF          <1>     call    print_msg
5684                              <1>
5685 000099DD E956F3FFFF          <1>     jmp     loc_file_rw_restore_retn
5686                              <1>
5687                              <1> ;loc_file_write_check_disk_space_err:
5688                              <1>     ;cmp    al, 27h ; Insufficient disk space
5689                              <1>     ;je     loc_file_write_insuff_disk_space_msg
5690                              <1>         ;jb loc_file_rw_cmd_failed
5691                              <1>
5692                              <1>     ;call  print_misc_error_msg ; 15/03/2016
5693                              <1>         ;jmp       loc_file_rw_restore_retn
5694                              <1>
5695                              <1> ; 19/12/2025
5696                              <1> %if 0
5697                              <1> change_fs_file_attributes:
5698                              <1>     ; 04/03/2016 ; Temporary
5699                              <1>     ; AL = File or directory attributes
5700                              <1>     ; AH = 0 -> Attributes are in MS-DOS format
5701                              <1>     ; AH > 0 -> Attributes are in SINGLIX format
5702                              <1>     ;push   ebx
5703                              <1>     ; ... do somethings here ...
5704                              <1>     ;pop    ebx
5705                              <1>     ; BL = File or directory attributes
5706                              <1>     retn
5707                              <1> %endif
5708                              <1>
5709                              <1> set_get_env:
5710                              <1>     ; 25/07/2022 - TRDOS 386 Kernel v2.0.5
5711                              <1>     ; 11/04/2016 (TRDOS 386 = TRDOS v2.0)
5712                              <1>     ; 02/09/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_set')
5713                              <1>     ; 2005 - 28/08/2011
5714                              <1> get_setenv_fchar:
5715                              <1>     ; esi = environment variable/string
5716 000099E2 8A06                <1>     mov     al, [esi]
5717 000099E4 3C20                <1>     cmp     al, 20h
5718 000099E6 771E                <1>     ja      short loc_find_env
5719                              <1>
5720 000099E8 BE00300900          <1>     mov     esi, Env_Page
5721                              <1> loc_print_setline:
5722 000099ED 803E00              <1>     cmp     byte [esi], 0
5723 000099F0 7613                <1>     jna     short loc_setenv_retn
5724 000099F2 E8B3D3FFFF          <1>     call    print_msg
5725 000099F7 56                  <1>     push    esi
5726 000099F8 BE[E7380100]        <1>     mov     esi, nextline
5727 000099FD E8A8D3FFFF          <1>     call    print_msg
5728 00009A02 5E                  <1>     pop     esi
5729 00009A03 EBE8                <1>     jmp     short loc_print_setline
5730                              <1>
5731                              <1> loc_setenv_retn:
5732 00009A05 C3                  <1>     retn
5733                              <1>
5734                              <1> loc_find_env:
5735 00009A06 3C3D                <1>     cmp     al, '='
5736                              <1>     ;je     loc_cmd_failed
5737                              <1>     ; 25/07/2022
5738 00009A08 7505                <1>     jne     short loc_find_envr
5739 00009A0A E935EAFFFF          <1>     jmp     loc_cmd_failed
5740                              <1>
5741                              <1> loc_find_envr:      ; 25/07/2022
5742 00009A0F 56                  <1>     push    esi
5743                              <1> loc_repeat_env_equal_check:
5744 00009A10 46                  <1>     inc     esi
5745 00009A11 803E3D              <1>     cmp     byte [esi], '='
5746 00009A14 7430                <1>     je      short pass_env_equal_check
5747 00009A16 803E20              <1>     cmp     byte [esi], 20h
5748 00009A19 73F5                <1>     jnb     short loc_repeat_env_equal_check
5749 00009A1B C60600              <1>     mov     byte [esi], 0
5750 00009A1E 5E                  <1>     pop     esi
5751                              <1>     ; 25/07/2022 (*)
5752                              <1> loc_print_env_string:
5753 00009A1F BF[42780100]        <1>     mov     edi, TextBuffer ; out buffer
5754 00009A24 B9FF000000          <1>     mov     ecx, 255 ; maximum size (limit)
5755 00009A29 30C0                <1>     xor     al, al ; 0 -> use [ESI]
5756 00009A2B E877000000          <1>     call    get_environment_string
5757 00009A30 72D3                <1>     jc      short loc_setenv_retn
5758                              <1>      ; 25/07/2022
5759                              <1> ;loc_print_env_string:
5760 00009A32 BE[42780100]        <1>     mov     esi, TextBuffer
5761 00009A37 E86ED3FFFF          <1>     call    print_msg
5762 00009A3C BE[E7380100]        <1>     mov     esi, nextline
5763                              <1>     ;call  print_msg
5764                              <1>     ;retn
5765                              <1>     ; 25/07/2022
5766 00009A41 E964D3FFFF          <1>     jmp     print_msg
5767                              <1>
5768                              <1> pass_env_equal_check:
5769 00009A46 46                  <1>     inc     esi
5770 00009A47 803E20              <1>     cmp     byte [esi], 20h
5771 00009A4A 73FA                <1>     jnb     short pass_env_equal_check
5772 00009A4C C60600              <1>     mov     byte [esi], 0
5773                              <1>
5774                              <1> loc_call_set_env_string:
5775 00009A4F 5E                  <1>     pop     esi
5776 00009A50 E815010000          <1>     call    set_environment_string
5777 00009A55 73AE                <1>     jnc     short loc_setenv_retn
5778                              <1>
5779                              <1> loc_set_cmd_failed:
5780 00009A57 3C08                <1>     cmp     al, 08h
5781                              <1>     ;jne    loc_cmd_failed
5782                              <1>     ; 25/07/2022
5783 00009A59 7405                <1>     je      short loc_set_cmd_failed_spc
5784 00009A5B E9E4E9FFFF          <1>     jmp     loc_cmd_failed
5785                              <1>
5786                              <1> loc_set_cmd_failed_spc:
5787 00009A60 BE[D2340100]        <1>     mov     esi, Msg_No_Set_Space
5788                              <1>     ;call  print_msg
5789                              <1>     ;retn
5790                              <1>     ; 25/07/2022
5791 00009A65 E940D3FFFF          <1>     jmp     print_msg
5792                              <1>
5793                              <1> set_get_path:
5794                              <1>     ; 25/07/2022 - TRDOS 386 Kernel v2.0.5
5795                              <1>     ; 11/04/2016 (TRDOS 386 = TRDOS v2.0)
5796                              <1>     ; 03/09/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_path')
5797                              <1>     ; 2005
5798                              <1> get_path_fchar:
5799                              <1>     ; esi = path
5800 00009A6A 803E20              <1>     cmp     byte [esi], 20h
5801 00009A6D 7711                <1>     ja      short loc_set_path
5802                              <1>
5803 00009A6F BE00300900          <1>     mov     esi, Env_Page
5804                              <1> loc_print_path:
5805 00009A74 803E00              <1>     cmp     byte [esi], 0
```

```
5806 00009A77 762D                <1>     jna     short loc_path_retn
5807                              <1>
5808 00009A79 BE[322F0100]        <1>     mov     esi, Cmd_Path ; 'PATH' address
5809                              <1>     ; 25/07/2022 (*)
5810 00009A7E EB9F                <1>     jmp     short loc_print_env_string
5811                              <1> ; 25/07/2022
5812                              <1> ;   mov     edi, TextBuffer ; out buffer
5813                              <1> ;   xor     al, al  ; use [ESI]
5814                              <1> ;   mov     ecx, 255 ; maximum size (limit)
5815                              <1> ;   call    get_environment_string
5816                              <1> ;   ;jc     short loc_path_retn
5817                              <1> ;   ; 25/07/2022
5818                              <1> ;   jnc     short loc_print_env_string
5819                              <1> ;   retn
5820                              <1>
5821                              <1> ;   mov     esi, TextBuffer
5822                              <1> ;   call    print_msg
5823                              <1> ;   mov     esi, nextline
5824                              <1> ;   ;call   print_msg
5825                              <1> ;loc_path_retn:
5826                              <1> ;   ;retn
5827                              <1> ;   ; 25/07/2022
5828                              <1> ;   jmp     print_msg
5829                              <1>
5830                              <1> loc_set_path:
5831 00009A80 56                  <1>     push    esi
5832                              <1> loc_set_path_find_end:
5833 00009A81 46                  <1>     inc     esi
5834 00009A82 803E20              <1>     cmp     byte [esi], 20h
5835 00009A85 73FA                <1>     jnb     short loc_set_path_find_end
5836 00009A87 C60600              <1>     mov     byte [esi], 0
5837                              <1> loc_set_path_header:
5838 00009A8A 5E                  <1>     pop     esi
5839                              <1> set_path_x: ; 31/12/2017 ('syspath')
5840 00009A8B 4E                  <1>     dec     esi
5841 00009A8C C6063D              <1>     mov     byte [esi], '='
5842 00009A8F 4E                  <1>     dec     esi
5843 00009A90 C60648              <1>     mov     byte [esi], 'H'
5844 00009A93 4E                  <1>     dec     esi
5845 00009A94 C60654              <1>     mov     byte [esi], 'T'
5846 00009A97 4E                  <1>     dec     esi
5847 00009A98 C60641              <1>     mov     byte [esi], 'A'
5848 00009A9B 4E                  <1>     dec     esi
5849 00009A9C C60650              <1>     mov     byte [esi], 'P'
5850                              <1>
5851                              <1> loc_path_call_set_env_string:
5852 00009A9F E8C6000000          <1>     call    set_environment_string
5853 00009AA4 72B1                <1>         jc  short loc_set_cmd_failed
5854                              <1> loc_path_retn:      ; 25/07/2022
5855 00009AA6 C3                  <1>     retn
5856                              <1>
5857                              <1> get_environment_string:
5858                              <1>     ; 12/04/2016
5859                              <1>     ; 11/04/2016
5860                              <1>     ; 05/04/2016 (TRDOS 386 = TRDOS v2.0)
5861                              <1>     ; 02/09/2011 (TRDOS v1, MAINPROG.ASM)
5862                              <1>     ; 28/08/2011
5863                              <1>     ; INPUT->
5864                              <1>     ;       EDI = Output buffer
5865                              <1>     ;       CX = Buffer length (<= ENV_PAGE_SIZE)
5866                              <1>     ;
5867                              <1>     ;       AL > 0 = AL = String sequence number
5868                              <1>     ;       AL = 0 -> ESI = ASCIIZ Set word
5869                              <1>     ;                 (environment variable)
5870                              <1>     ; OUTPUT ->
5871                              <1>     ;       ESI is not changed
5872                              <1>     ;       EDI is not changed
5873                              <1>     ;       EAX = String length (with zero tail)
5874                              <1>     ;       EDX = Environment variables page address
5875                              <1>     ;       CF = 1 -> Not found (EAX not valid)
5876                              <1>     ;
5877                              <1>     ; (Modified registers: EAX, EDX)
5878                              <1>
5879 00009AA7 BA00300900          <1>     mov     edx, Env_Page
5880 00009AAC 803A00              <1>     cmp     byte [edx], 0
5881 00009AAF 7474                <1>     jz      short get_env_string_with_word_stc_retn
5882                              <1>
5883 00009AB1 66890D[AE820100]    <1>     mov     [env_var_length], cx
5884                              <1>
5885 00009AB8 51                  <1>     push    ecx ; *
5886 00009AB9 56                  <1>     push    esi ; **
5887                              <1>
5888 00009ABA 08C0                <1>     or      al, al
5889 00009ABC 7449                <1>     jz      short get_env_string_with_word
5890                              <1>
5891                              <1> get_env_string_with_seq_number:
5892 00009ABE B101                <1>     mov     cl, 1
5893 00009AC0 88C5                <1>     mov     ch, al
5894 00009AC2 31C0                <1>     xor     eax, eax
5895 00009AC4 89D6                <1>     mov     esi, edx ; Env_Page
5896                              <1>
5897                              <1> get_env_string_seq_number_check:
5898 00009AC6 38CD                <1>     cmp     ch, cl
5899 00009AC8 7726                <1>     ja      short get_env_string_seq_number_next
5900                              <1>
5901                              <1> get_env_string_move_to_buff:
5902 00009ACA 57                  <1>     push    edi ; ***
5903                              <1>
5904 00009ACB 29D2                <1>     sub     edx, edx
5905                              <1>
5906                              <1> get_env_string_seq_number_repeat1:
5907 00009ACD 42                  <1>     inc     edx
5908 00009ACE AC                  <1>     lodsb
5909 00009ACF AA                  <1>     stosb
5910                              <1>
5911 00009AD0 66FF0D[AE820100]    <1>     dec     word [env_var_length]
5912 00009AD7 7508                <1>     jnz     short get_env_string_seq_number_repeat3
5913                              <1>
5914                              <1> get_env_string_seq_number_repeat2:
5915 00009AD9 20C0                <1>     and     al, al
5916 00009ADB 7408                <1>     jz      short get_env_string_seq_number_ok
5917 00009ADD 42                  <1>     inc     edx
5918 00009ADE AC                  <1>     lodsb
5919 00009ADF EBF8                <1>     jmp     short get_env_string_seq_number_repeat2
5920                              <1>
5921                              <1> get_env_string_seq_number_repeat3:
5922 00009AE1 08C0                <1>     or      al, al
5923 00009AE3 75E8                <1>     jnz     short get_env_string_seq_number_repeat1
5924                              <1>
5925                              <1> get_env_string_seq_number_ok:
5926 00009AE5 5F                  <1>     pop     edi ; ***
5927 00009AE6 89D0                <1>     mov     eax, edx ; Length of the environment string
5928                              <1>                      ; (ASCIIZ, includes ZERO tail)
5929 00009AE8 BA00300900          <1>     mov     edx, Env_Page
```

```
5930                                <1>
5931                                <1> get_env_string_stc_retn:
5932 00009AED 5E                   <1>     pop     esi ; **
5933 00009AEE 59                   <1>     pop     ecx ; *
5934 00009AEF C3                   <1>     retn
5935                                <1>
5936                                <1> get_env_string_seq_number_next:
5937 00009AF0 AC                   <1>     lodsb
5938 00009AF1 08C0                 <1>     or      al, al
5939 00009AF3 75FB                 <1>     jnz     short get_env_string_seq_number_next
5940                                <1>
5941 00009AF5 81FE00320900         <1>     cmp     esi, Env_Page + Env_Page_Size ; +512 (+4096)
5942 00009AFB F5                   <1>     cmc
5943 00009AFC 72EF                 <1>     jc      short get_env_string_stc_retn
5944                                <1>
5945 00009AFE AC                   <1>     lodsb
5946 00009AFF 3C01                 <1>     cmp     al, 1
5947 00009B01 72EA                 <1>     jb      short get_env_string_stc_retn
5948 00009B03 FEC1                 <1>     inc     cl
5949 00009B05 EBBF                 <1>     jmp     short get_env_string_seq_number_check
5950                                <1>
5951                                <1> get_env_string_with_word:
5952 00009B07 31C9                 <1>     xor     ecx, ecx
5953                                <1>
5954                                <1> get_env_string_calc_word_length:
5955 00009B09 AC                   <1>     lodsb
5956 00009B0A 3C20                 <1>     cmp     al, 20h
5957 00009B0C 7211                 <1>     jb      short get_env_string_calc_word_length_ok
5958                                <1>     ;inc    cx
5959 00009B0E FEC1                 <1>     inc     cl
5960                                <1>
5961 00009B10 3C61                 <1>     cmp     al, 'a'
5962 00009B12 72F5                 <1>     jb      short get_env_string_calc_word_length
5963 00009B14 3C7A                 <1>     cmp     al, 'z'
5964 00009B16 77F1                 <1>     ja      short get_env_string_calc_word_length
5965 00009B18 24DF                 <1>     and     al, 0DFh
5966 00009B1A 8846FF               <1>     mov     [esi-1], al
5967 00009B1D EBEA                 <1>     jmp     short get_env_string_calc_word_length
5968                                <1>
5969                                <1> get_env_string_calc_word_length_ok:
5970 00009B1F 08C9                 <1>     or      cl, cl
5971 00009B21 7506                 <1>     jnz     short get_env_string_calc_word_length_save
5972                                <1>
5973 00009B23 5E                   <1>     pop     esi ; **
5974                                <1>
5975                                <1> get_env_string_stc_retn1:
5976 00009B24 59                   <1>     pop     ecx ; *
5977                                <1>
5978                                <1> get_env_string_with_word_stc_retn:
5979 00009B25 31C0                 <1>     xor     eax, eax
5980 00009B27 F9                   <1>     stc
5981 00009B28 C3                   <1>     retn
5982                                <1>
5983                                <1> get_env_string_calc_word_length_save:
5984 00009B29 871C24               <1>     xchg    ebx, [esp] ; **
5985 00009B2C 89DE                 <1>     mov     esi, ebx
5986                                <1>             ; Start of the env string (to be searched)
5987                                <1>
5988 00009B2E 57                   <1>     push    edi ; ***
5989 00009B2F 89D7                 <1>     mov     edi, edx ; Env_Page
5990                                <1>
5991                                <1> get_env_string_compare:
5992 00009B31 57                   <1>     push    edi ; ****
5993 00009B32 51                   <1>     push    ecx ; ***** ; Variable name length
5994                                <1>
5995                                <1> get_env_string_compare_rep:
5996 00009B33 AC                   <1>     lodsb
5997 00009B34 AE                   <1>     scasb
5998 00009B35 7511                 <1>     jne     short get_env_string_compare_next1
5999 00009B37 E2FA                 <1>     loop    get_env_string_compare_rep
6000                                <1>
6001 00009B39 803F3D               <1>     cmp     byte [edi], '='
6002 00009B3C 750A                 <1>     jne     short get_env_string_compare_next1
6003                                <1>
6004 00009B3E 59                   <1>     pop     ecx ; *****
6005 00009B3F 5F                   <1>     pop     edi ; ****
6006 00009B40 89FE                 <1>     mov     esi, edi
6007 00009B42 5F                   <1>     pop     edi ; ***
6008 00009B43 871C24               <1>     xchg    ebx, [esp] ; **
6009 00009B46 EB82                 <1>     jmp     short get_env_string_move_to_buff
6010                                <1>
6011                                <1> get_env_string_compare_next1:
6012 00009B48 89FE                 <1>     mov     esi, edi
6013 00009B4A 59                   <1>     pop     ecx ; *****
6014 00009B4B 5F                   <1>     pop     edi ; ****
6015                                <1> get_env_string_compare_next2:
6016 00009B4C 81FEFF310900         <1>     cmp     esi, Env_Page + Env_Page_Size - 1 ; +511 (+4095)
6017 00009B52 7310                 <1>     jnb     short get_env_string_compare_not_ok
6018 00009B54 20C0                 <1>     and     al, al
6019 00009B56 AC                   <1>     lodsb
6020 00009B57 75F3                 <1>     jnz     short get_env_string_compare_next2
6021 00009B59 08C0                 <1>     or      al, al
6022 00009B5B 7407                 <1>     jz      short get_env_string_compare_not_ok
6023 00009B5D 4E                   <1>     dec     esi ; 12/04/2016
6024 00009B5E 89F7                 <1>     mov     edi, esi
6025 00009B60 89DE                 <1>     mov     esi, ebx
6026 00009B62 EBCD                 <1>     jmp     short get_env_string_compare
6027                                <1>
6028                                <1> get_env_string_compare_not_ok:
6029 00009B64 5F                   <1>     pop     edi ; ***
6030 00009B65 89DE                 <1>     mov     esi, ebx
6031 00009B67 5B                   <1>     pop     ebx ; **
6032 00009B68 EBBA                 <1>     jmp     short get_env_string_stc_retn1
6033                                <1>
6034                                <1> set_environment_string:
6035                                <1>     ; 25/07/2022 - TRDOS 386 Kernel v2.0.5
6036                                <1>     ; 13/04/2016
6037                                <1>     ; 12/04/2016
6038                                <1>     ; 11/04/2016
6039                                <1>     ; 06/04/2016
6040                                <1>     ; 05/04/2016 (TRDOS 386 = TRDOS v2.0)
6041                                <1>     ; 02/09/2011 (TRDOS v1, MAINPROG.ASM)
6042                                <1>     ; 29/08/2011
6043                                <1>     ; 29/08/2011
6044                                <1>     ; INPUT->
6045                                <1>     ;       ESI = ASCIIZ environment string
6046                                <1>     ; OUTPUT ->
6047                                <1>     ;       ESI is not changed
6048                                <1>     ;       CF = 1 -> Could not set,
6049                                <1>     ;            insufficient environment space
6050                                <1>     ;
6051                                <1>     ; (EAX, EDX will be changed)
6052                                <1>     ;
6053                                <1>     ;    (EAX = Start address of the env string if > 0)
```

```
                                <1>     ;    (EDX = Environment string length)
                                <1>
00009B6A 56                     <1>     push    esi ; *
                                <1>
00009B6B 31C0                   <1>     xor     eax, eax
                                <1>
                                <1> set_env_chk_validation1:
00009B6D FEC4                   <1>     inc     ah ; variable (string) length
00009B6F AC                     <1>     lodsb
00009B70 3C3D                   <1>     cmp     al, '='
00009B72 7415                   <1>     je      short set_env_chk_validation2
00009B74 3C20                   <1>     cmp     al, 20h
00009B76 720F                   <1>     jb      short set_env_string_stc
                                <1>
                                <1>     ; 06/04/2016
00009B78 3C61                   <1>     cmp     al, 'a'
00009B7A 72F1                   <1>     jb      short set_env_chk_validation1
00009B7C 3C7A                   <1>     cmp     al, 'z'
00009B7E 77ED                   <1>     ja      short set_env_chk_validation1
00009B80 2C20                   <1>     sub     al, 'a'-'A'
00009B82 8846FF                 <1>     mov     [esi-1], al
00009B85 EBE6                   <1>     jmp     short set_env_chk_validation1
                                <1>
                                <1> set_env_string_stc:
00009B87 5E                     <1>     pop     esi ; *
                                <1>     ;stc
00009B88 C3                     <1>     retn
                                <1>
                                <1> set_env_chk_validation2:
00009B89 51                     <1>     push    ecx ; **
00009B8A 53                     <1>     push    ebx ; ***
00009B8B 57                     <1>     push    edi ; ****
                                <1>
                                <1>     ; 12/04/2016
                                <1>     ;mov    ebx, [esp+12]
                                <1>     ; 25/07/2022
00009B8C 8B54240C               <1>     mov     edx, [esp+12]
                                <1>
                                <1> set_env_chk_validation2w:
00009B90 89F7                   <1>     mov     edi, esi
00009B92 4F                     <1>     dec     edi
                                <1>
00009B93 807FFF20               <1>     cmp     byte [edi-1], 20h
00009B97 771A                   <1>     ja      short set_env_chk_validation2z
                                <1>
00009B99 56                     <1>     push    esi
00009B9A 89FE                   <1>     mov     esi, edi
00009B9C 4E                     <1>     dec     esi
                                <1>
                                <1> set_env_chk_validation2x:
00009B9D 4E                     <1>     dec     esi
                                <1>
                                <1>     ;cmp    esi, ebx
00009B9E 39D6                   <1>     cmp     esi, edx ; 25/07/2022
00009BA0 7207                   <1>     jb      short set_env_chk_validation2y
                                <1>
00009BA2 4F                     <1>     dec     edi
                                <1>
00009BA3 8A06                   <1>     mov     al, [esi]
00009BA5 8807                   <1>     mov     [edi], al
                                <1>
00009BA7 EBF4                   <1>     jmp     short set_env_chk_validation2x
                                <1>
                                <1> set_env_chk_validation2y:
00009BA9 5E                     <1>     pop     esi
                                <1>
                                <1>     ;;mov   byte [ebx], 20h
                                <1>     ; 25/07/2022
                                <1>     ;mov    byte [edx], 20h
                                <1>
                                <1>     ;inc    ebx
                                <1>     ;mov    [esp+12], ebx
                                <1>     ; 25/07/2022
00009BAA 42                     <1>     inc     edx
00009BAB 8954240C               <1>     mov     [esp+12], edx
                                <1>
00009BAF FECC                   <1>     dec     ah ; 13/04/2016
                                <1>
00009BB1 EBDD                   <1>     jmp     short set_env_chk_validation2w
                                <1>
                                <1> set_env_chk_validation2z:
                                <1>     ;mov    edx, Env_Page
                                <1>     ;mov    edi, edx
                                <1>     ; 25/07/2022
00009BB3 BB00300900             <1>     mov     ebx, Env_Page
00009BB8 89DF                   <1>     mov     edi, ebx
                                <1>
                                <1> set_env_chk_validation3:
00009BBA AC                     <1>     lodsb
00009BBB 3C20                   <1>     cmp     al, 20h
00009BBD 74FB                   <1>     je      short set_env_chk_validation3
                                <1>
00009BBF 9C                     <1>     pushf
                                <1>
                                <1>     ; 12/04/2016
                                <1> set_env_chk_validation3n:
00009BC0 3C61                   <1>     cmp     al, 'a'
00009BC2 720C                   <1>     jb      short set_env_chk_validation3c
00009BC4 3C7A                   <1>     cmp     al, 'z'
00009BC6 7705                   <1>     ja      short set_env_chk_validation3x
00009BC8 2C20                   <1>     sub     al, 'a'-'A'
00009BCA 8846FF                 <1>     mov     [esi-1], al
                                <1>
                                <1> set_env_chk_validation3x:
00009BCD AC                     <1>     lodsb
00009BCE EBF0                   <1>     jmp     short set_env_chk_validation3n
                                <1>
                                <1> set_env_chk_validation3c:
00009BD0 3C20                   <1>     cmp     al, 20h
00009BD2 73F9                   <1>     jnb     short set_env_chk_validation3x
                                <1>
00009BD4 803F00                 <1>     cmp     byte [edi], 0
00009BD7 772B                   <1>     ja      short set_env_chk_validation4
                                <1>
00009BD9 9D                     <1>     popf
00009BDA 7222                   <1>     jb      short set_env_string_nothing
                                <1>
00009BDC B900020000             <1>     mov     ecx, Env_Page_Size ; 512 (4096)
                                <1>
                                <1>     ;mov    esi, ebx ; 12/04/2016
                                <1>     ; 25/07/2022
00009BE1 89D6                   <1>     mov     esi, edx
                                <1>
                                <1>     ; 25/07/2022
```

```
6178 00009BE3 89CA              <1>     mov     edx, ecx
6179                            <1>
6180                            <1> set_env_string_copy_to_envb:
6181 00009BE5 AC                <1>     lodsb
6182 00009BE6 3C20              <1>     cmp     al, 20h
6183 00009BE8 7207              <1>     jb      short set_env_string_copy_to_envb_z
6184 00009BEA AA                <1>     stosb
6185 00009BEB E2F8              <1>     loop    set_env_string_copy_to_envb
6186                            <1>
6187                            <1>     ; 11/04/2016
6188                            <1>     ;mov    edi, edx ; Env_Page
6189                            <1>     ; 25/07/2022
6190 00009BED 89DF              <1>     mov     edi, ebx
6191                            <1>     ; 25/07/2022
6192                            <1>     ;mov    ecx, Env_Page_Size
6193 00009BEF 89D1              <1>     mov     ecx, edx
6194                            <1>
6195                            <1> set_env_string_copy_to_envb_z:
6196                            <1>     ; 25/07/2022
6197                            <1>     ;push   edx  ; Start address of the variable
6198                            <1>
6199                            <1>     ;;mov   edx, Env_Page_Size
6200                            <1>     ;; 25/07/2022
6201 00009BF1 29CA              <1>     sub     edx, ecx ; variable (string) length
6202                            <1>
6203 00009BF3 28C0              <1>     sub     al, al ; 0
6204 00009BF5 F3AA              <1>     rep     stosb ; clear remain bytes of the env page
6205                            <1>
6206                            <1>     ;pop    eax  ; Start address of the variable
6207                            <1>     ; 25/07/2022
6208 00009BF7 89D8              <1>     mov     eax, ebx
6209                            <1>
6210                            <1> set_env_string_allocate_envb_retn:  ; stc or clc return
6211 00009BF9 5F                <1>     pop     edi ; ****
6212 00009BFA 5B                <1>     pop     ebx ; ***
6213 00009BFB 59                <1>     pop     ecx ; **
6214 00009BFC 5E                <1>     pop     esi ; *
6215 00009BFD C3                <1>     retn
6216                            <1>
6217                            <1> set_env_string_nothing:
6218 00009BFE 31C0              <1>     xor     eax, eax
6219 00009C00 31D2              <1>     xor     edx, edx ; 11/04/2016
6220 00009C02 EBF5              <1>     jmp     short set_env_string_allocate_envb_retn
6221                            <1>
6222                            <1> set_env_chk_validation4:
6223                            <1>     ; 11/04/2016
6224 00009C04 9D                <1>     popf
6225                            <1>
6226                            <1>     ;mov    esi, edx  ; Env_Page
6227                            <1>     ; 25/07/2022
6228 00009C05 89DE              <1>     mov     esi, ebx
6229                            <1>
6230                            <1> set_env_chk_validation5:
6231                            <1>     ;mov    edi, ebx ; ASCIIZ environment string address
6232                            <1>     ; 25/07/2022
6233 00009C07 89D7              <1>     mov     edi, edx
6234 00009C09 0FB6CC            <1>     movzx   ecx, ah ; Variable (string) length (with '=')
6235                            <1>
6236                            <1> set_env_chk_validation5_loop:
6237 00009C0C AC                <1>     lodsb
6238 00009C0D AE                <1>     scasb
6239 00009C0E 7508              <1>     jne     short set_env_chk_validation6
6240 00009C10 E2FA              <1>     loop    set_env_chk_validation5_loop
6241                            <1>
6242 00009C12 3C3D              <1>     cmp     al, '='
6243                            <1>         ;je set_env_change_variable
6244                            <1>     ; 25/07/2022
6245 00009C14 7502              <1>     jne     short set_env_chk_validation6
6246 00009C16 EB7E              <1>     jmp     set_env_change_variable
6247                            <1>
6248                            <1> set_env_chk_validation6:
6249 00009C18 08C0              <1>     or      al, al ; 0
6250 00009C1A 7403              <1>     jz      short set_env_chk_validation7
6251                            <1>
6252 00009C1C AC                <1>     lodsb
6253 00009C1D EBF9              <1>     jmp     short set_env_chk_validation6
6254                            <1>
6255                            <1> set_env_chk_validation7:
6256 00009C1F 88E1              <1>     mov     cl, ah
6257 00009C21 01F1              <1>     add     ecx, esi
6258 00009C23 81F9FF310900      <1>     cmp     ecx, Env_Page + Env_Page_Size - 1
6259                            <1>             ; 511 (4095)
6260                            <1>             ; strlen + '=' + 0
6261 00009C29 72DC              <1>     jb      short set_env_chk_validation5
6262                            <1>
6263                            <1> set_env_chk_validation8: ; variable not found
6264 00009C2B 0FB6F4            <1>     movzx   esi, ah  ; variable name length (with '=')
6265                            <1>     ;add    esi, ebx ; position just after of the '='
6266                            <1>     ; 25/07/2022
6267 00009C2E 01D6              <1>     add     esi, edx
6268                            <1>
6269                            <1> set_env_chk_validation8_loop:
6270 00009C30 AC                <1>     lodsb
6271 00009C31 3C20              <1>     cmp     al, 20h
6272 00009C33 74FB              <1>     je      short set_env_chk_validation8_loop
6273 00009C35 72C7              <1>     jb      short set_env_string_nothing
6274                            <1>
6275                            <1> set_env_chk_validation9:
6276 00009C37 AC                <1>     lodsb
6277 00009C38 3C20              <1>     cmp     al, 20h
6278 00009C3A 73FB              <1>     jnb     short set_env_chk_validation9
6279                            <1>
6280                            <1>     ; End of ASCIIZ environment string
6281                            <1>
6282                            <1> set_env_add_variable:
6283                            <1>     ;sub    esi, ebx ; variable+definition length
6284                            <1>     ; 25/07/2022
6285 00009C3C 29D6              <1>     sub     esi, edx
6286                            <1>
6287 00009C3E 56                <1>     push    esi ; *****
6288                            <1>
6289                            <1>     ;mov    esi, edx ; Environment page address
6290                            <1>     ; 25/07/2022
6291 00009C3F 89DE              <1>     mov     esi, ebx
6292                            <1>
6293 00009C41 B900020000        <1>     mov     ecx, Env_Page_Size ; 512 (4096)
6294                            <1>
6295                            <1> set_env_add_variable_loop:
6296 00009C46 AC                <1>     lodsb
6297 00009C47 20C0              <1>     and     al, al
6298 00009C49 7406              <1>     jz      short set_env_add_variable_chk1 ; 0
6299 00009C4B E2F9              <1>     loop    set_env_add_variable_loop
6300                            <1>
6301                            <1>     ; 11/04/2016
```

```
6302 00009C4D 884EFF              <1>     mov     [esi-1], cl ; 0
6303 00009C50 41                  <1>     inc     ecx
6304                              <1>
6305                              <1> set_env_add_variable_chk1:
6306 00009C51 49                  <1>     dec     ecx
6307 00009C52 7408                <1>     jz      short set_env_add_variable_nspc
6308 00009C54 AC                  <1>     lodsb
6309 00009C55 08C0                <1>     or      al, al
6310 00009C57 740B                <1>     jz      short set_env_add_variable_chk2 ; 00
6311 00009C59 49                  <1>     dec     ecx
6312 00009C5A 75EA                <1>     jnz     short set_env_add_variable_loop
6313                              <1>
6314                              <1> set_env_add_variable_nspc: ; no space on environment page
6315 00009C5C 58                  <1>     pop     eax ; *****
6316                              <1>     ;mov    eax, 8 ; No space for new environment string
6317                              <1>     ; 25/07/2022
6318 00009C5D 29C0                <1>     sub     eax, eax
6319 00009C5F B008                <1>     mov     al, 8
6320 00009C61 F9                  <1>     stc
6321 00009C62 EB95                <1>         jmp     short set_env_string_allocate_envb_retn
6322                              <1>
6323                              <1> set_env_add_variable_chk2:
6324 00009C64 8B0C24              <1>     mov     ecx, [esp] ; *****
6325 00009C67 4E                  <1>     dec     esi ; beginning address of the new variable
6326 00009C68 89F0                <1>     mov     eax, esi
6327 00009C6A 01C8                <1>     add     eax, ecx ; string length (with CR)
6328                              <1>     ;add    edx, Env_Page_Size ; 512 (4096)
6329                              <1>     ; 25/07/2022
6330 00009C6C 81C300020000        <1>     add     ebx, Env_Page_Size
6331                              <1>     ;cmp    eax, edx
6332 00009C72 39D8                <1>     cmp     eax, ebx ; 25/07/2022
6333 00009C74 77E6                <1>     ja      short set_env_add_variable_nspc
6334 00009C76 49                  <1>     dec     ecx ; except CR at the end
6335                              <1>     ;mov    edx, ecx ; 12/04/2016
6336                              <1>     ; 25/07/2022
6337 00009C77 89CB                <1>     mov     ebx, ecx
6338 00009C79 89F7                <1>     mov     edi, esi
6339 00009C7B 893C24              <1>     mov     [esp], edi ; ***** ; Start address of new variable
6340                              <1>     ;mov    esi, ebx ; ASCIIZ environment string address
6341                              <1>     ; 25/07/2022
6342 00009C7E 89D6                <1>     mov     esi, edx
6343 00009C80 F3A4                <1>     rep     movsb
6344 00009C82 28C0                <1>     sub     al, al
6345 00009C84 AA                  <1>     stosb
6346 00009C85 58                  <1>     pop     eax ; ***** ; Beginning address of new variable
6347 00009C86 81FF00320900        <1>         cmp     edi, Env_Page + Env_Page_Size ; 12/04/2016
6348                              <1>     ;jnb    set_env_string_allocate_envb_retn ; OK !
6349                              <1>     ; 25/07/2022
6350 00009C8C 7303                <1>     jnb     short set_env_add_variable_chk3
6351 00009C8E 880F                <1>     mov     [edi], cl ; 0
6352 00009C90 F8                  <1>     clc     ; 13/04/2016
6353                              <1> set_env_add_variable_chk3:
6354 00009C91 E963FFFFFF          <1>         jmp     set_env_string_allocate_envb_retn ; OK !
6355                              <1>
6356                              <1> set_env_change_variable:
6357                              <1>     ; 06/04/2016
6358                              <1>     ; esi = Variable's address in environment page (after '=')
6359                              <1>     ; edi = ASCIIZ environment string address (after '=')
6360                              <1>
6361                              <1>     ; ah = variable length from start to the '='
6362 00009C96 8825[AE820100]      <1>     mov     [env_var_length], ah
6363                              <1>
6364 00009C9C 28C9                <1>     sub     cl, cl ; ecx = 0
6365                              <1>
6366 00009C9E 57                  <1>     push    edi ; *****
6367                              <1>
6368 00009C9F 89F7                <1>     mov     edi, esi ; 11/04/2016
6369                              <1>
6370                              <1> set_env_change_variable_calc1:
6371 00009CA1 AC                  <1>     lodsb
6372 00009CA2 08C0                <1>     or      al, al
6373 00009CA4 7403                <1>     jz      short set_env_change_variable_calc2
6374                              <1>
6375 00009CA6 41                  <1>     inc     ecx ; length of environment string (after the '=')
6376                              <1>
6377 00009CA7 EBF8                <1>     jmp     short set_env_change_variable_calc1
6378                              <1>
6379                              <1> set_env_change_variable_calc2:
6380 00009CA9 8B3424              <1>     mov     esi, [esp] ; ASCIIZ environment string address
6381                              <1>
6382 00009CAC 29D2                <1>     sub     edx, edx
6383                              <1>
6384                              <1> set_env_change_variable_calc3:
6385 00009CAE AC                  <1>     lodsb
6386 00009CAF 3C20                <1>     cmp     al, 20h
6387 00009CB1 7203                <1>     jb      short set_env_change_variable_calc4
6388                              <1>
6389 00009CB3 42                  <1>     inc     edx ; length of ASCIIZ string (after the '=')
6390                              <1>
6391 00009CB4 EBF8                <1>     jmp     short set_env_change_variable_calc3
6392                              <1>
6393                              <1> set_env_change_variable_calc4:
6394 00009CB6 C646FF00            <1>     mov     byte [esi-1], 0  ; put ZERO instead of CR
6395                              <1>
6396 00009CBA 5E                  <1>     pop     esi ; ***** ; ASCIIZ string address (after '=')
6397                              <1>
6398                              <1>     ; EDI = Old variable's address (after '=')
6399                              <1>
6400                              <1>     ; compare the new string with the old string
6401 00009CBB 39CA                <1>     cmp     edx, ecx
6402 00009CBD 7718                <1>     ja      short set_env_change_variable_calc5 ; longer
6403                              <1>     ;jb     set_env_change_variable_calc9 ; shorter
6404                              <1>     ; 25/07/2022
6405 00009CBF 7405                <1>     je      short set_env_change_variable_calc22
6406 00009CC1 E98C000000          <1>     jmp     set_env_change_variable_calc9
6407                              <1>
6408                              <1> set_env_change_variable_calc22:
6409                              <1>     ;same length (simple copy)
6410 00009CC6 0FB6C4              <1>     movzx   eax, ah
6411 00009CC9 01C2                <1>     add     edx, eax
6412 00009CCB F7D8                <1>     neg     eax
6413 00009CCD 01F8                <1>     add     eax, edi
6414                              <1>     ; EAX = Start address of the variable
6415                              <1>     ; EDX = Variable length (without ZERO at the end of variable)
6416                              <1>
6417 00009CCF F3A4                <1>     rep     movsb
6418 00009CD1 F8                  <1>     clc     ; 13/04/2016
6419 00009CD2 E922FFFFFF          <1>         jmp     set_env_string_allocate_envb_retn ; OK !
6420                              <1>
6421                              <1> set_env_change_variable_calc5:
6422                              <1>     ; 11/04/2016
6423 00009CD7 52                  <1>     push    edx ; *****
6424 00009CD8 29CA                <1>     sub     edx, ecx ; difference ; (the new string is longer)
6425 00009CDA 89F3                <1>     mov     ebx, esi
```

```
6426 00009CDC 89FE                <1>     mov     esi, edi
6427                              <1>
6428                              <1> set_env_change_variable_calc6:
6429 00009CDE AC                  <1>     lodsb
6430 00009CDF 20C0                <1>     and     al, al
6431 00009CE1 75FB                <1>     jnz     short set_env_change_variable_calc6
6432                              <1>
6433 00009CE3 81FE00320900        <1>     cmp     esi, Env_Page + Env_Page_Size ; 512 (4096)
6434 00009CE9 0F836DFFFFFF        <1>         jnb     set_env_add_variable_nspc
6435                              <1>
6436 00009CEF 89F9                <1>     mov     ecx, edi  ; current (old) variable's address
6437 00009CF1 89F7                <1>     mov     edi, esi  ; next variable's address
6438                              <1>
6439 00009CF3 AC                  <1>     lodsb
6440 00009CF4 08C0                <1>     or      al, al
6441 00009CF6 7417                <1>     jz      short set_env_change_variable_calc8 ; 00
6442                              <1>
6443                              <1> set_env_change_variable_calc7:
6444 00009CF8 AC                  <1>     lodsb
6445 00009CF9 20C0                <1>     and     al, al
6446 00009CFB 75FB                <1>     jnz     short set_env_change_variable_calc7
6447                              <1>
6448 00009CFD 81FE00320900        <1>     cmp     esi, Env_Page + Env_Page_Size ; 512 (4096)
6449                              <1>     ;jnb    set_env_add_variable_nspc
6450                              <1>     ; 25/07/2022
6451 00009D03 7205                <1>     jb      short set_env_change_variable_calc24
6452                              <1> set_env_change_variable_calc23:
6453 00009D05 E952FFFFFF          <1>     jmp     set_env_add_variable_nspc
6454                              <1>
6455                              <1> set_env_change_variable_calc24:
6456 00009D0A AC                  <1>     lodsb
6457 00009D0B 08C0                <1>     or      al, al
6458 00009D0D 75E9                <1>     jnz     short set_env_change_variable_calc7
6459                              <1>
6460                              <1> set_env_change_variable_calc8:
6461 00009D0F 4E                  <1>     dec     esi ; address of the second (last) 0 of the 00
6462                              <1>
6463 00009D10 01F2                <1>     add     edx, esi ; final position of the last 0
6464                              <1>
6465 00009D12 81FA00320900        <1>     cmp     edx, Env_Page + Env_Page_Size ; 512 (4096)
6466                              <1>     ;jnb     set_env_add_variable_nspc
6467                              <1>     ; 25/07/2022
6468 00009D18 73EB                <1>     jnb     short set_env_change_variable_calc23
6469                              <1>
6470 00009D1A 89C8                <1>     mov     eax, ecx ; old variable's address (after '=')
6471                              <1>
6472 00009D1C 89F1                <1>     mov     ecx, esi
6473 00009D1E 29F9                <1>     sub     ecx, edi ; count of bytes to move forward
6474                              <1>
6475                              <1>     ; 13/04/2016
6476 00009D20 C60200              <1>     mov     byte [edx], 0
6477 00009D23 89D7                <1>     mov     edi, edx
6478 00009D25 29F2                <1>     sub     edx, esi ; difference (additional byte count)
6479 00009D27 4F                  <1>     dec     edi ; the last zero address (first byte of the 00)
6480 00009D28 89FE                <1>     mov     esi, edi
6481 00009D2A 29D6                <1>     sub     esi, edx ; - displacement
6482                              <1>
6483 00009D2C FA                  <1>     cli     ; disable interrupts
6484 00009D2D FD                  <1>     std     ; backward
6485                              <1>
6486 00009D2E F3A4                <1>     rep     movsb ; move ECX bytes from DS:ESI to ES:EDI
6487                              <1>
6488 00009D30 FC                  <1>     cld     ; forward (default)
6489 00009D31 FB                  <1>     sti     ; enable interrupts
6490                              <1>
6491 00009D32 89C7                <1>     mov     edi, eax
6492 00009D34 59                  <1>     pop     ecx ; ***** ; byte count (after '=')
6493 00009D35 89CA                <1>     mov     edx, ecx
6494 00009D37 89DE                <1>     mov     esi, ebx ; ASCIIZ string address (after '=')
6495 00009D39 89FB                <1>     mov     ebx, edi
6496                              <1>
6497 00009D3B F3A4                <1>     rep     movsb
6498                              <1>
6499 00009D3D 880F                <1>     mov     [edi], cl ; 0 ; end of variable
6500                              <1>
6501 00009D3F 0FB605[AE820100]    <1>     movzx   eax, byte [env_var_length]
6502 00009D46 01C2                <1>     add     edx, eax ; variable length (total)
6503 00009D48 F7D8                <1>     neg     eax
6504 00009D4A 01D8                <1>     add     eax, ebx ; start address of the variable
6505 00009D4C F8                  <1>     clc     ; 13/04/2016
6506 00009D4D E9A7FEFFFF          <1>         jmp     set_env_string_allocate_envb_retn ; OK !
6507                              <1>
6508                              <1> set_env_change_variable_calc9:
6509                              <1>     ; 11/04/2016
6510 00009D52 21D2                <1>     and     edx, edx ; is empty ?
6511 00009D54 753B                <1>     jnz     short set_env_change_variable_calc15
6512                              <1>
6513 00009D56 0FB6DC              <1>     movzx   ebx, ah
6514 00009D59 F7DB                <1>     neg     ebx
6515 00009D5B 01FB                <1>     add     ebx, edi
6516                              <1>
6517                              <1>     ; EBX = Start address of the variable (in env page)
6518                              <1>     ; EDX = Variable length = 0
6519                              <1>
6520 00009D5D 89FE                <1>     mov     esi, edi
6521                              <1>
6522                              <1> set_env_change_variable_calc10:
6523 00009D5F AC                  <1>     lodsb
6524 00009D60 08C0                <1>     or      al, al
6525 00009D62 75FB                <1>     jnz     short set_env_change_variable_calc10
6526                              <1>
6527 00009D64 B9FF310900          <1>     mov     ecx, Env_Page + Env_Page_Size - 1
6528                              <1>
6529 00009D69 39CE                <1>     cmp     esi, ecx ; +511 (+4095)
6530 00009D6B 7604                <1>     jna     short set_env_change_variable_calc11
6531                              <1>
6532 00009D6D 89CE                <1>     mov     esi, ecx
6533 00009D6F 8806                <1>     mov     [esi], al ; 0
6534                              <1>
6535                              <1> set_env_change_variable_calc11:
6536 00009D71 89DF                <1>     mov     edi, ebx ; old variable's start address
6537                              <1>
6538                              <1> set_env_change_variable_calc12:
6539 00009D73 AC                  <1>     lodsb
6540 00009D74 AA                  <1>     stosb
6541 00009D75 20C0                <1>     and     al, al
6542 00009D77 75FA                <1>     jnz     short set_env_change_variable_calc12
6543 00009D79 39CE                <1>     cmp     esi, ecx
6544 00009D7B 7706                <1>     ja      short set_env_change_variable_calc13
6545 00009D7D AC                  <1>     lodsb
6546 00009D7E AA                  <1>     stosb
6547 00009D7F 20C0                <1>     and     al, al
6548 00009D81 75F0                <1>     jnz     short set_env_change_variable_calc12
6549                              <1>
```

```
6550                            <1> set_env_change_variable_calc13:
6551 00009D83 29F9             <1>     sub     ecx, edi
6552 00009D85 7203             <1>     jb      short set_env_change_variable_calc14
6553 00009D87 41               <1>     inc     ecx ; 1-512 (1-4096)
6554 00009D88 F3AA             <1>     rep     stosb ; al = 0
6555                            <1>
6556                            <1> set_env_change_variable_calc14:
6557 00009D8A 29C0             <1>     sub     eax, eax ; Start address of the variable
6558                            <1>     ; EAX = 0 -> Variable is removed
6559                            <1>     ; EDX = Variable length = 0
6560                            <1>
6561 00009D8C E968FEFFFF       <1>         jmp     set_env_string_allocate_envb_retn ; OK !
6562                            <1>
6563                            <1> set_env_change_variable_calc15:
6564 00009D91 52               <1>     push    edx ; *****
6565 00009D92 F7DA             <1>     neg     edx
6566 00009D94 01CA             <1>     add     edx, ecx ; difference (the old string is longer)
6567 00009D96 89F3             <1>     mov     ebx, esi
6568 00009D98 89FE             <1>     mov     esi, edi
6569                            <1>
6570                            <1> set_env_change_variable_calc16:
6571 00009D9A AC               <1>     lodsb
6572 00009D9B 20C0             <1>     and     al, al
6573 00009D9D 75FB             <1>     jnz     short set_env_change_variable_calc16
6574                            <1>
6575 00009D9F B900320900       <1>     mov     ecx, Env_Page + Env_Page_Size
6576                            <1>
6577 00009DA4 39CE             <1>     cmp     esi, ecx ; +512 (+4096)
6578 00009DA6 7605             <1>     jna     short set_env_change_variable_calc17
6579                            <1>
6580 00009DA8 89CE             <1>     mov     esi, ecx
6581 00009DAA 8846FF           <1>     mov     [esi-1], al ; 0
6582                            <1>
6583                            <1> set_env_change_variable_calc17:
6584 00009DAD 89F9             <1>     mov     ecx, edi  ; current (old) variable's address
6585 00009DAF 89F7             <1>     mov     edi, esi  ; next variable's address
6586                            <1>
6587 00009DB1 AC               <1>     lodsb
6588 00009DB2 08C0             <1>     or      al, al
6589 00009DB4 741D             <1>     jz      short set_env_change_variable_calc20
6590                            <1>
6591                            <1> set_env_change_variable_calc18:
6592 00009DB6 AC               <1>     lodsb
6593 00009DB7 20C0             <1>     and     al, al
6594 00009DB9 75FB             <1>     jnz     short set_env_change_variable_calc18
6595                            <1>
6596 00009DBB 81FE00320900     <1>     cmp     esi, Env_Page + Env_Page_Size
6597 00009DC1 720B             <1>     jb      short set_env_change_variable_calc19
6598 00009DC3 740E             <1>     je      short set_env_change_variable_calc20
6599                            <1>
6600 00009DC5 BEFF310900       <1>     mov     esi, Env_Page + Env_Page_Size - 1
6601 00009DCA 8806             <1>     mov     [esi], al ; 0
6602 00009DCC EB06             <1>     jmp     short set_env_change_variable_calc21
6603                            <1>
6604                            <1> set_env_change_variable_calc19:
6605 00009DCE AC               <1>     lodsb
6606 00009DCF 08C0             <1>     or      al, al
6607 00009DD1 75E3             <1>     jnz     short set_env_change_variable_calc18
6608                            <1>
6609                            <1> set_env_change_variable_calc20:
6610 00009DD3 4E               <1>     dec     esi ; address of the second (last) 0 of the 00
6611                            <1>
6612                            <1> set_env_change_variable_calc21:
6613                            <1>     ; edx = difference (byte count)
6614                            <1>
6615 00009DD4 89C8             <1>     mov     eax, ecx ; old variable's address (after '=')
6616                            <1>
6617 00009DD6 89F1             <1>     mov     ecx, esi
6618 00009DD8 29F9             <1>     sub     ecx, edi ; count of bytes to move backward
6619                            <1>
6620 00009DDA 89FE             <1>     mov     esi, edi ; next variable's address
6621 00009DDC 29D7             <1>     sub     edi, edx ; (displacement)
6622                            <1>
6623 00009DDE F3A4             <1>     rep     movsb
6624                            <1>
6625 00009DE0 880F             <1>     mov     [edi], cl ; 0 ; 00 ; end of environment variables
6626                            <1>
6627 00009DE2 89C7             <1>     mov     edi, eax
6628 00009DE4 5A               <1>     pop     edx ; ***** ; byte count (after '=')
6629 00009DE5 89D1             <1>     mov     ecx, edx
6630 00009DE7 89DE             <1>     mov     esi, ebx ; ASCIIZ string address (after '=')
6631 00009DE9 89FB             <1>     mov     ebx, edi
6632                            <1>
6633 00009DEB F3A4             <1>     rep     movsb
6634                            <1>
6635 00009DED 880F             <1>     mov     [edi], cl ; 0 ; end of variable
6636                            <1>
6637 00009DEF 0FB605[AE820100] <1>     movzx   eax, byte [env_var_length]
6638 00009DF6 01C2             <1>     add     edx, eax ; variable length (total)
6639 00009DF8 F7D8             <1>     neg     eax
6640 00009DFA 01D8             <1>     add     eax, ebx ; start address of the variable
6641 00009DFC F8               <1>     clc     ; 13/04/2016
6642 00009DFD E9F7FDFFFF       <1>         jmp     set_env_string_allocate_envb_retn ; OK !
6643                            <1>
6644                            <1>     ; 19/12/2025
6645                            <1> loc_load_mainprog_cfg_exit:
6646 00009E02 C3               <1>     retn
6647                            <1>
6648                            <1> mainprog_startup_configuration:
6649                            <1>     ; 19/12/2025 - TRDOS 386 Kernel v2.10.10
6650                            <1>     ; 25/07/2022 - TRDOS 386 Kernel v2.0.5
6651                            <1>     ; 22/11/2017
6652                            <1>     ; 06/05/2016
6653                            <1>     ; 14/04/2016 (TRDOS 386 = TRDOS v2.0)
6654                            <1>     ; 17/09/2011 (TRDOS v1, MAINPROG.ASM)
6655                            <1>     ;
6656                            <1> loc_load_mainprog_cfg_file:
6657 00009E03 BE[AC2E0100]     <1>     mov     esi, MainProgCfgFile
6658 00009E08 66B80018         <1>     mov     ax, 1800h ; Except volume label and dirs
6659 00009E0C E859EBFFFF       <1>     call    find_first_file
6660 00009E11 72EF             <1>     jc      short loc_load_mainprog_cfg_exit
6661                            <1>
6662                            <1>     ;or     eax, eax
6663                            <1>     ;jz     short loc_load_mainprog_cfg_exit
6664                            <1>
6665                            <1> loc_start_mainprog_configuration:
6666                            <1>     ; ESI = FindFile_DirEntry Location
6667                            <1>     ; EAX = File Size
6668                            <1>
6669 00009E13 A3[30770100]     <1>     mov     [MainProgCfg_FileSize], eax
6670                            <1>
6671 00009E18 668B5614         <1>     mov     dx, [esi+DirEntry_FstClusHI]
6672 00009E1C C1E210           <1>     shl     edx, 16
6673 00009E1F 668B561A         <1>     mov     dx, [esi+DirEntry_FstClusLO]
```

```
                                <1>     ;mov    [csftdf_sf_cluster], edx
                                <1>     ; 19/12/2025
00009E23 8915[64820100]         <1>     mov     [MainProgCfg_cluster], edx
                                <1>
00009E29 89C1                   <1>     mov     ecx, eax
00009E2B 29C0                   <1>     sub     eax, eax
                                <1>
                                <1>     ; TRDOS 386 (TRDOS v2.0)
                                <1>     ; Allocate contiguous memory block for loading the file
                                <1>
                                <1>     ; eax = 0 (Allocate memory from the beginning)
                                <1>     ; ecx = File (Allocation) size in bytes
                                <1>
00009E2D E807BFFFFF             <1>     call    allocate_memory_block
00009E32 72CE                   <1>     jc      short loc_load_mainprog_cfg_exit
                                <1>
                                <1>     ;mov    [csftdf_sf_mem_addr], eax ; loading address
                                <1>     ;mov    [csftdf_sf_mem_bsize], ecx ; block size
                                <1>     ; 19/12/2025
00009E34 A3[5C820100]           <1>     mov     [MainProgCfg_mem_addr], eax ; loading address
00009E39 890D[60820100]         <1>     mov     [MainProgCfg_mem_bsize], ecx ; block size
                                <1>
00009E3F 31DB                   <1>     xor     ebx, ebx
                                <1>     ;;mov   [csftdf_sf_rbytes], ebx ; 0, reset
                                <1>     ; 19/12/2025
                                <1>     ;mov    [MainProgCfg_rbytes], ebx ; 0, reset
                                <1>
00009E41 8A3D[42770100]         <1>     mov     bh, [Current_Drv] ; [FindFile_Drv]
00009E47 BE00010900             <1>     mov     esi, Logical_DOSDisks
00009E4C 01DE                   <1>     add     esi, ebx
                                <1>
                                <1>     ;mov    ebx, [csftdf_sf_mem_addr] ; memory block address
                                <1>     ; 19/12/2025
00009E4E 8B1D[5C820100]         <1>     mov     ebx, [MainProgCfg_mem_addr]
                                <1>
                                <1> ; 19/12/2025
                                <1> %if 0
                                <1>     cmp     byte [esi+LD_FATType], 0
                                <1>         ja  short loc_mcfg_load_fat_file
                                <1>
                                <1>     ;mov    dword [csftdf_r_size], 65536
                                <1>     ; 19/12/2025
                                <1>     mov     dword [MainProgCfg_r_size], 65536
                                <1>         jmp     loc_mcfg_load_fs_file
                                <1>
                                <1> loc_load_mainprog_cfg_exit:
                                <1>     retn
                                <1> %endif
                                <1>
                                <1> loc_mcfg_load_fat_file:
                                <1>     ;movzx  eax, word [esi+LD_BPB+BytesPerSec]
                                <1>     ; 19/12/2025
00009E54 B800020000             <1>     mov     eax, 512
00009E59 0FB64E13               <1>     movzx   ecx, byte [esi+LD_BPB+SecPerClust]
00009E5D F7E1                   <1>     mul     ecx
                                <1>     ;mov    [csftdf_r_size], eax
                                <1>     ; 19/12/2025
00009E5F A3[6C820100]           <1>     mov     [MainProgCfg_r_size], eax
                                <1>
                                <1> loc_mcfg_load_fat_file_next:
00009E64 E817010000             <1>     call    mcfg_read_fat_file_sectors
00009E69 7259                   <1>         jc  short mcfg_deallocate_mem ; 25/07/2022
                                <1>
00009E6B 09D2                   <1>     or      edx, edx ; edx > 0 -> EOF
00009E6D 74F5                   <1>     jz      short loc_mcfg_load_fat_file_next
                                <1>
                                <1> loc_mcfg_load_fat_file_ok:
                                <1>     ; 06/05/2016
00009E6F C705[00830100]-        <1>     mov     dword [mainprog_return_addr], loc_mcfg_ci_return_addr
00009E75 [419F0000]             <1>
                                <1>     ;
                                <1>     ;mov    esi, [csftdf_sf_mem_addr]
                                <1>     ; 19/12/2025
00009E79 8B35[5C820100]         <1>     mov     esi, [MainProgCfg_mem_addr]
00009E7F 8935[34770100]         <1>     mov     [MainProgCfg_LineOffset], esi
                                <1>
00009E85 A1[30770100]           <1>     mov     eax, [MainProgCfg_FileSize]
00009E8A 89C2                   <1>     mov     edx, eax
00009E8C 01F2                   <1>     add     edx, esi
                                <1>
                                <1> loc_mcfg_process_next_line_check:
00009E8E 89C1                   <1>     mov     ecx, eax
                                <1>
00009E90 803E2A                 <1>     cmp     byte [esi], "*" ; Remark sign
00009E93 7503                   <1>     jne     short loc_mcfg_process_next_line
00009E95 46                     <1>     inc     esi
00009E96 EB16                   <1>     jmp     short loc_move_mainprog_cfg_nl1
                                <1>
                                <1> loc_mcfg_process_next_line:
00009E98 83F94F                 <1>     cmp     ecx, 79
00009E9B 7604                   <1>     jna     short loc_start_mainprog_cfg_process
                                <1>
                                <1>     ;mov    ecx, 79
                                <1>     ; 25/07/2022
00009E9D 29C9                   <1>     sub     ecx, ecx
00009E9F B14F                   <1>     mov     cl, 79
                                <1>
                                <1> loc_start_mainprog_cfg_process:
00009EA1 BF[F2770100]           <1>     mov     edi, CommandBuffer
                                <1>
                                <1> loc_move_mainprog_cfg_line:
00009EA6 AC                     <1>     lodsb
00009EA7 3C20                   <1>     cmp     al, 20h
00009EA9 720C                   <1>     jb      short loc_move_mainprog_cfg_nl2
00009EAB AA                     <1>     stosb
00009EAC E2F8                   <1>     loop    loc_move_mainprog_cfg_line
                                <1>
                                <1> loc_move_mainprog_cfg_nl1:
00009EAE 39D6                   <1>     cmp     esi, edx ; + configuration file size
00009EB0 7322                   <1>     jnb     short loc_end_of_mainprog_cfg_line
00009EB2 AC                     <1>     lodsb
00009EB3 3C20                   <1>     cmp     al, 20h
00009EB5 73F7                   <1>     jnb     short loc_move_mainprog_cfg_nl1
                                <1>
                                <1> loc_move_mainprog_cfg_nl2:
00009EB7 39D6                   <1>     cmp     esi, edx
00009EB9 7319                   <1>     jnb     short loc_end_of_mainprog_cfg_line
00009EBB 8A06                   <1>     mov     al, [esi]
00009EBD 3C20                   <1>     cmp     al, 20h
00009EBF 7713                   <1>     ja      short loc_end_of_mainprog_cfg_line
00009EC1 46                     <1>     inc     esi
00009EC2 EBF3                   <1>     jmp     short loc_move_mainprog_cfg_nl2
                                <1>
                                <1>     ; 25/07/2022
```

```
                                <1> mcfg_deallocate_mem:
                                <1>     ;mov    eax, [csftdf_sf_mem_addr] ; start address
                                <1>     ;mov    ecx, [csftdf_sf_mem_bsize] ; block size
                                <1>     ; 19/12/2025
00009EC4 A1[5C820100]           <1>     mov     eax, [MainProgCfg_mem_addr] ; start address
00009EC9 8B0D[60820100]         <1>     mov     ecx, [MainProgCfg_mem_bsize] ; block size
                                <1>
                                <1>     ;call   deallocate_memory_block
                                <1>     ;retn
00009ECF E97FC0FFFF             <1>     jmp     deallocate_memory_block
                                <1>
                                <1> loc_end_of_mainprog_cfg_line:
00009ED4 C60700                 <1>     mov     byte [edi], 0
                                <1>
00009ED7 8935[34770100]         <1>     mov     [MainProgCfg_LineOffset], esi
                                <1>
                                <1>     ; 22/11/2017
00009EDD BE[FA770100]           <1>     mov     esi, CommandBuffer + 8
00009EE2 29FE                   <1>     sub     esi, edi
00009EE4 7606                   <1>     jna     short loc_move_mainprog_cfg_command
00009EE6 30C0                   <1>     xor     al, al
                                <1> loc_mainprog_cfg_clear_chrs:
00009EE8 AA                     <1>     stosb
00009EE9 4E                     <1>     dec     esi
00009EEA 75FC                   <1>     jnz     short loc_mainprog_cfg_clear_chrs
                                <1>
                                <1> loc_move_mainprog_cfg_command:
00009EEC BE[F2770100]           <1>     mov     esi, CommandBuffer
00009EF1 89F7                   <1>     mov     edi, esi
00009EF3 31DB                   <1>     xor     ebx, ebx
                                <1>     ;xor    ecx, ecx
00009EF5 30C9                   <1>     xor     cl, cl
                                <1>
                                <1> loc_move_mcfg_first_cmd_char:
00009EF7 8A041E                 <1>     mov     al, [esi+ebx]
00009EFA FEC3                   <1>     inc     bl
00009EFC 3C20                   <1>     cmp     al, 20h
00009EFE 7712                   <1>     ja      short loc_move_mcfg_cmd_capitalizing
00009F00 7237                   <1>     jb      short loc_move_mcfg_cmd_arguments_ok
00009F02 80FB4F                 <1>     cmp     bl, 79
00009F05 72F0                   <1>     jb      short loc_move_mcfg_first_cmd_char
00009F07 EB30                   <1>     jmp     short loc_move_mcfg_cmd_arguments_ok
                                <1>
                                <1> loc_move_mcfg_next_cmd_char:
00009F09 8A041E                 <1>     mov     al, [esi+ebx]
00009F0C FEC3                   <1>     inc     bl
00009F0E 3C20                   <1>     cmp     al, 20h
00009F10 7614                   <1>     jna     short loc_move_mcfg_cmd_ok
                                <1>
                                <1> loc_move_mcfg_cmd_capitalizing:
00009F12 3C61                   <1>     cmp     al, 61h ; 'a'
00009F14 7206                   <1>     jb      short loc_move_mcfg_cmd_caps_ok
00009F16 3C7A                   <1>     cmp     al, 7Ah ; 'z'
00009F18 7702                   <1>     ja      short loc_move_mcfg_cmd_caps_ok
00009F1A 24DF                   <1>     and     al, 0DFh ; sub al, 'a'-'A'
                                <1>
                                <1> loc_move_mcfg_cmd_caps_ok:
00009F1C AA                     <1>     stosb
00009F1D FEC1                   <1>     inc     cl
00009F1F 80FB4F                 <1>     cmp     bl, 79
00009F22 72E5                   <1>     jb      short loc_move_mcfg_next_cmd_char
00009F24 EB13                   <1>     jmp     short loc_move_mcfg_cmd_arguments_ok
                                <1>
                                <1> loc_move_mcfg_cmd_ok:
00009F26 30C0                   <1>     xor     al, al ; 0
                                <1>
                                <1> loc_move_mcfg_cmd_arguments:
00009F28 8807                   <1>     mov     [edi], al
00009F2A 47                     <1>     inc     edi
00009F2B 80FB4F                 <1>     cmp     bl, 79
00009F2E 7309                   <1>     jnb     short loc_move_mcfg_cmd_arguments_ok
00009F30 8A041E                 <1>     mov     al, [esi+ebx]
00009F33 FEC3                   <1>     inc     bl
00009F35 3C20                   <1>     cmp     al, 20h
00009F37 73EF                   <1>     jnb     short loc_move_mcfg_cmd_arguments
                                <1>
                                <1> loc_move_mcfg_cmd_arguments_ok:
00009F39 C60700                 <1>     mov     byte [edi], 0
                                <1>
                                <1> loc_mcfg_process_cmd_interpreter:
00009F3C E828E1FFFF             <1>     call    command_interpreter
                                <1>
                                <1> loc_mcfg_ci_return_addr:
00009F41 A1[30770100]           <1>     mov     eax, [MainProgCfg_FileSize]
00009F46 89C2                   <1>     mov     edx, eax
00009F48 8B35[34770100]         <1>     mov     esi, [MainProgCfg_LineOffset]
00009F4E 01F2                   <1>     add     edx, esi
                                <1>     ;add    eax, [csftdf_sf_mem_addr]
                                <1>     ; 19/12/2025
00009F50 0305[5C820100]         <1>     add     eax, [MainProgCfg_mem_addr]
00009F56 29F0                   <1>     sub     eax, esi
00009F58 0F8730FFFFFF           <1>         ja      loc_mcfg_process_next_line_check
                                <1>
00009F5E E861FFFFFF             <1>     call    mcfg_deallocate_mem
                                <1>
00009F63 B94F000000             <1>     mov     ecx, 79 ; 80 ?
00009F68 BF[F2770100]           <1>     mov     edi, CommandBuffer
00009F6D 30C0                   <1>     xor     al, al
00009F6F F3AA                   <1>     rep     stosb
                                <1>
                                <1>     ; 06/05/2016
00009F71 BE[E7380100]           <1>     mov     esi, nextline
00009F76 E82FCEFFFF             <1>     call    print_msg
00009F7B E98FD8FFFF             <1>     jmp     dos_prompt
                                <1>
                                <1>     ; 05/12/2025
                                <1> ;mcfg_read_file_sectors:
                                <1>     ; 14/04/2016
                                <1>     ;cmp    byte [esi+LD_FATType], 0
                                <1>         ;jna       short mcfg_read_fs_file_sectors
                                <1>
                                <1> mcfg_read_fat_file_sectors:
                                <1>     ; return:
                                <1>     ;   CF = 0 & EDX > 0 -> END OF FILE
                                <1>     ;   CF = 0 & EDX = 0 -> not EOF
                                <1>     ;   CF = 1 -> read error (error code in AL)
                                <1>
                                <1> mcfg_read_fat_file_secs_0:
00009F80 8B15[30770100]         <1>     mov     edx, [MainProgCfg_FileSize]
                                <1>     ;sub    edx, [csftdf_sf_rbytes]
                                <1>     ; 19/12/2025
00009F86 2B15[74820100]         <1>     sub     edx, [MainProgCfg_rbytes]
                                <1>     ;cmp    edx, [csftdf_r_size]
00009F8C 3B15[6C820100]         <1>     cmp     edx, [MainProgCfg_r_size]
```

```
6921 00009F92 7306              <1>     jnb     short mcfg_read_fat_file_secs_1
6922                            <1>     ;mov    [csftdf_r_size], edx
6923                            <1>     ; 19/12/2025
6924 00009F94 8915[6C820100]    <1>     mov     [MainProgCfg_r_size], edx
6925                            <1>
6926                            <1> mcfg_read_fat_file_secs_1:
6927                            <1>     ;mov    eax, [csftdf_r_size]
6928                            <1>     ; 19/12/2025
6929 00009F9A A1[6C820100]      <1>     mov     eax, [MainProgCfg_r_size]
6930 00009F9F 29D2              <1>     sub     edx, edx
6931                            <1>     ;movzx  ecx, word [esi+LD_BPB+BytesPerSec]
6932                            <1>     ; 19/12/2025
6933 00009FA1 B900020000        <1>     mov     ecx, 512
6934 00009FA6 01C8              <1>     add     eax, ecx
6935 00009FA8 48                <1>     dec     eax
6936 00009FA9 F7F1              <1>     div     ecx
6937 00009FAB 89C1              <1>     mov     ecx, eax ; sector count
6938                            <1>     ;mov    eax, [csftdf_sf_cluster]
6939                            <1>     ; 19/12/2025
6940 00009FAD A1[64820100]      <1>     mov     eax, [MainProgCfg_cluster]
6941                            <1>
6942                            <1>     ; EBX = memory block address (current)
6943                            <1>
6944 00009FB2 E8B8210000        <1>     call    read_fat_file_sectors
6945 00009FB7 7230              <1>     jc      short mcfg_read_fat_file_secs_3
6946                            <1>
6947                            <1>     ; EBX = next memory address
6948                            <1>
6949                            <1>     ;mov    eax, [csftdf_sf_rbytes]
6950                            <1>     ; 19/12/2025
6951 00009FB9 A1[74820100]      <1>     mov     eax, [MainProgCfg_rbytes]
6952                            <1>     ;add    eax, [csftdf_r_size]
6953 00009FBE 0305[6C820100]    <1>     add     eax, [MainProgCfg_r_size]
6954 00009FC4 8B15[30770100]    <1>     mov     edx, [MainProgCfg_FileSize]
6955 00009FCA 39D0              <1>     cmp     eax, edx
6956 00009FCC 731B              <1>     jnb     short mcfg_read_fat_file_secs_3 ; edx > 0
6957                            <1>     ;mov    [csftdf_sf_rbytes], eax
6958                            <1>     ; 19/12/2025
6959 00009FCE A3[74820100]      <1>     mov     [MainProgCfg_rbytes], eax
6960                            <1>
6961 00009FD3 53                <1>     push    ebx ; *
6962                            <1>     ; get next cluster (csftdf_r_size! bytes)
6963                            <1>     ;mov    eax, [csftdf_sf_cluster]
6964                            <1>     ; 19/12/2025
6965 00009FD4 A1[64820100]      <1>     mov     eax, [MainProgCfg_cluster]
6966 00009FD9 E89B1F0000        <1>     call    get_next_cluster
6967 00009FDE 5B                <1>     pop     ebx ; *
6968 00009FDF 7301              <1>     jnc     short mcfg_read_fat_file_secs_2
6969                            <1>
6970                            <1>     ;mov    eax, 17; Read error !
6971 00009FE1 C3                <1>     retn
6972                            <1>
6973                            <1> mcfg_read_fat_file_secs_2:
6974 00009FE2 29D2              <1>     sub     edx, edx ; 0
6975                            <1>     ;mov    [csftdf_sf_cluster], eax ; next cluster
6976                            <1>     ; 19/12/2025
6977 00009FE4 A3[64820100]      <1>     mov     [MainProgCfg_cluster], eax
6978                            <1>
6979                            <1> ; 25/07/2022 - TRDOS 386 Kernel v2.0.5
6980                            <1>
6981                            <1> mcfg_read_fat_file_secs_3:
6982                            <1> ; 19/12/2025
6983                            <1> ;   ;retn
6984                            <1> ;
6985                            <1> ;mcfg_read_fs_file_sectors:
6986                            <1> ;   ;retn
6987                            <1> ;
6988                            <1> ;loc_mcfg_load_fs_file:
6989 00009FE9 C3                <1>     retn
6990                            <1>
6991                            <1> load_and_execute_file:
6992                            <1>     ; 02/12/2025
6993                            <1>     ; 04/07/2025
6994                            <1>     ; 03/07/2025 - TRDOS 386 Kernel v2.0.10
6995                            <1>     ; 03/09/2024 - TRDOS 386 Kernel v2.0.9
6996                            <1>     ; 30/08/2023 - TRDOS 386 Kernel v2.0.6
6997                            <1>     ; 25/07/2022 - TRDOS 386 Kernel v2.0.5
6998                            <1>     ; 04/01/2017
6999                            <1>     ; 06/05/2016 - 07/05/2016 - 11/05/2016
7000                            <1>     ; 23/04/2016 - 24/04/2016
7001                            <1>     ; 22/04/2016 (TRDOS 386 = TRDOS v2.0)
7002                            <1>     ; 05/11/2011
7003                            <1>     ; (TRDOS v1, CMDINTR.ASM, 'cmp_cmd_run', 'cmp_cmd_external')
7004                            <1>     ; ('loc_run_check_filename')
7005                            <1>     ; 29/08/2011
7006                            <1>     ; 10/09/2011
7007                            <1>     ; INPUT->
7008                            <1>     ;       ESI = Path Name address (CommandBuffer address)
7009                            <1>     ; OUTPUT ->
7010                            <1>     ;       none (error message will be shown if an error will occur)
7011                            <1>     ;
7012                            <1>     ; (EAX, EBX, ECX, EDX, ESI, EDI, EBP will be changed)
7013                            <1>     ;
7014                            <1> loc_run_check_filename:
7015 00009FEA 803E20            <1>     cmp     byte [esi], 20h
7016                            <1>     ;jb     loc_cmd_failed
7017                            <1>     ; 25/07/2022
7018 00009FED 7237              <1>     jb      short loc_run_ppn_failed
7019 00009FEF 7703              <1>     ja      short loc_run_check_filename_ok
7020 00009FF1 46                <1>     inc     esi
7021 00009FF2 EBF6              <1>     jmp     short loc_run_check_filename
7022                            <1>
7023                            <1> loc_run_check_filename_ok:
7024 00009FF4 C605[A3770100]00  <1>     mov     byte [CmdArgStart], 0 ; reset
7025 00009FFB 56                <1>     push    esi ; *
7026                            <1> loc_run_get_first_arg_pos:
7027 00009FFC 46                <1>     inc     esi
7028 00009FFD 8A06              <1>     mov     al, [esi]
7029 00009FFF 3C20              <1>     cmp     al, 20h
7030 0000A001 77F9              <1>     ja      short loc_run_get_first_arg_pos
7031 0000A003 C60600            <1>     mov     byte [esi], 0
7032                            <1> loc_run_get_external_arg_pos:
7033                            <1>     ; 11/05/2016
7034 0000A006 46                <1>     inc     esi
7035 0000A007 8A06              <1>     mov     al, [esi]
7036 0000A009 3C20              <1>     cmp     al, 20h
7037 0000A00B 760C              <1>     jna     short loc_run_parse_path_name
7038 0000A00D 89F0              <1>     mov     eax, esi
7039 0000A00F 2D[F2770100]      <1>     sub     eax, CommandBuffer
7040 0000A014 A2[A3770100]      <1>     mov     byte [CmdArgStart], al
7041                            <1> loc_run_parse_path_name:
7042 0000A019 5E                <1>     pop     esi ; *
7043 0000A01A BF[E27F0100]      <1>     mov     edi, FindFile_Drv
7044 0000A01F E84A090000        <1>     call    parse_path_name
```

```
7045                                    <1>     ;jc    loc_cmd_failed
7046                                    <1>     ; 25/07/2022
7047 0000A024 7305                      <1>     jnc    short loc_run_check_filename_exists
7048                                    <1> loc_run_ppn_failed:
7049 0000A026 E919E4FFFF                <1>     jmp    loc_cmd_failed
7050                                    <1>
7051                                    <1> loc_run_check_filename_exists:
7052 0000A02B BE[24800100]              <1>     mov    esi, FindFile_Name
7053 0000A030 803E20                    <1>     cmp    byte [esi], 20h
7054                                    <1>     ;jna   loc_cmd_failed
7055                                    <1>     ; 25/07/2022
7056 0000A033 76F1                      <1>     jna    short loc_run_ppn_failed
7057                                    <1>
7058                                    <1> loc_run_check_exe_filename_ext:
7059 0000A035 E896020000                <1>     call   check_prg_filename_ext
7060                                    <1>     ;jc    loc_cmd_failed
7061                                    <1>     ; 25/07/2022
7062 0000A03A 72EA                      <1>     jc     short loc_run_ppn_failed
7063                                    <1>     ; 03/07/2025
7064                                    <1>     ; if al = 0 -> ah = file name length
7065                                    <1>
7066                                    <1> loc_run_check_exe_filename_ext_ok:
7067 0000A03C 66A3[FD820100]            <1>     mov    word [EXE_ID], ax ; 'P.' or 'P'+0
7068                                    <1>
7069                                    <1> loc_run_drv:
7070 0000A042 C605[FC820100]00          <1>     mov    byte [Run_Manual_Path], 0
7071 0000A049 A1[3C770100]              <1>     mov    eax, [Current_Dir_FCluster]
7072 0000A04E A3[F0820100]              <1>         mov     [Run_CDirFC], eax
7073                                    <1>     ;
7074 0000A053 8A35[42770100]            <1>     mov    dh, [Current_Drv]
7075 0000A059 8835[9F7E0100]            <1>     mov    [RUN_CDRV], dh
7076                                    <1>
7077 0000A05F 8A15[E27F0100]            <1>     mov    dl, [FindFile_Drv]
7078 0000A065 38F2                      <1>     cmp    dl, dh
7079 0000A067 7413                      <1>     je     short loc_run_change_directory
7080                                    <1>
7081 0000A069 8005[FC820100]02          <1>     add    byte [Run_Manual_Path], 2
7082                                    <1>
7083 0000A070 E86BD6FFFF                <1>     call   change_current_drive
7084                                    <1>     ;jc    loc_run_cmd_failed
7085                                    <1>     ; 25/07/2022
7086 0000A075 7305                      <1>     jnc    short loc_run_change_directory
7087 0000A077 E9F3E3FFFF                <1>     jmp    loc_run_cmd_failed
7088                                    <1>
7089                                    <1> loc_run_change_directory:
7090 0000A07C 803D[E37F0100]20          <1>     cmp    byte [FindFile_Directory], 20h
7091 0000A083 7624                      <1>     jna    short loc_run_find_executable_file
7092                                    <1>
7093 0000A085 FE05[FC820100]            <1>     inc    byte [Run_Manual_Path]
7094                                    <1>
7095 0000A08B FE05[5D2E0100]            <1>     inc    byte [Restore_CDIR]
7096                                    <1>
7097 0000A091 BE[E37F0100]              <1>     mov    esi, FindFile_Directory
7098 0000A096 30E4                      <1>     xor    ah, ah ; CD_COMMAND sign -> 0
7099 0000A098 E83E030000                <1>     call   change_current_directory
7100                                    <1>     ;jc    loc_run_cmd_failed
7101                                    <1>     ; 25/07/2022
7102 0000A09D 7305                      <1>     jnc    short loc_run_change_prompt_dir_string
7103 0000A09F E9CBE3FFFF                <1>     jmp    loc_run_cmd_failed
7104                                    <1>
7105                                    <1> loc_run_change_prompt_dir_string:
7106 0000A0A4 E85B020000                <1>     call   change_prompt_dir_string
7107                                    <1>
7108                                    <1> loc_run_find_executable_file:
7109 0000A0A9 66C705[F8820100]00-       <1>     mov    word [Run_Auto_Path], 0
7109 0000A0B1 00                        <1>
7110                                    <1>
7111                                    <1> loc_run_find_executable_file_next:
7112 0000A0B2 BE[24800100]              <1>     mov    esi, FindFile_Name
7113                                    <1> loc_run_find_program_file_next:
7114 0000A0B7 66B80018                  <1>     mov    ax, 1800h ; Except volume label and dirs
7115 0000A0BB E8AAE8FFFF                <1>     call   find_first_file
7116                                    <1>     ; ESI = Directory Entry (FindFile_DirEntry) Location
7117                                    <1>     ; EDI = Directory Buffer Directory Entry Location
7118                                    <1>     ; EAX = File size
7119                                    <1>     ;jnc   loc_load_and_run_file
7120                                    <1>     ; 25/07/2022
7121 0000A0C0 7216                      <1>     jc     short loc_run_program_file_not_found
7122                                    <1>
7123                                    <1>     ; 02/12/2025 (TRDOS 386 v2.0.10)
7124 0000A0C2 8B471C                    <1>     mov    eax, [edi+DirEntry_FileSize]
7125 0000A0C5 21C0                      <1>     and    eax, eax ; zero file size check?
7126 0000A0C7 750A                      <1>     jnz    short loc_load_and_run_file_@
7127 0000A0C9 B814000000                <1>     mov    eax, ERR_FILE_SIZE
7128                                    <1>            ; 20, 'file size error !'
7129                                    <1>     ;mov   eax, ERR_ZERO_LENGTH
7130                                    <1>            ; 20, 'zero length !' error
7131 0000A0CE E99CE3FFFF                <1>     jmp    loc_run_cmd_failed
7132                                    <1>
7133                                    <1> loc_load_and_run_file_@:
7134 0000A0D3 E957010000                <1>     jmp    loc_load_and_run_file
7135                                    <1>
7136                                    <1> loc_run_program_file_not_found:
7137 0000A0D8 3C02                      <1>     cmp    al, 2 ; file not found
7138                                    <1>     ;jne   loc_run_cmd_failed
7139                                    <1>     ; 25/07/2022
7140 0000A0DA 7405                      <1>     je     short loc_run_progr_file_chk_prg_ext
7141 0000A0DC E98EE3FFFF                <1>     jmp    loc_run_cmd_failed
7142                                    <1>
7143                                    <1> loc_run_progr_file_chk_prg_ext: ; 25/07/2022
7144 0000A0E1 66A1[FD820100]            <1>     mov    ax, word [EXE_ID]
7145                                    <1>
7146                                    <1>     ;cmp   ah, '.' ; File name has extension sign
7147                                    <1>     ;je    short loc_run_check_auto_path
7148                                    <1>     ;or    al, al
7149                                    <1>     ;jnz   short loc_run_check_auto_path
7150                                    <1>     ; 03/07/2025
7151 0000A0E7 663D502E                  <1>     cmp    ax, 'P.'
7152 0000A0EB 7420                      <1>     je     short loc_run_check_auto_path
7153                                    <1>
7154 0000A0ED 80FC08                    <1>     cmp    ah, 8 ; count of file name chars
7155 0000A0F0 771B                      <1>     ja     short loc_run_check_auto_path
7156                                    <1>
7157                                    <1> loc_run_change_file_ext_to_prg:
7158 0000A0F2 0FB6DC                    <1>     movzx  ebx, ah ; count of file name chars
7159 0000A0F5 BE[24800100]              <1>     mov    esi, FindFile_Name
7160 0000A0FA 01F3                      <1>     add    ebx, esi
7161                                    <1>     ; 07/05/2016
7162 0000A0FC C7032E505247              <1>     mov    dword [ebx],  '.PRG'
7163 0000A102 66C705[FD820100]50-       <1>     mov    word [EXE_ID], 'P.'
7163 0000A10A 2E                        <1>
7164 0000A10B EBAA                      <1>     jmp    short loc_run_find_program_file_next
7165                                    <1>
7166                                    <1> loc_run_check_auto_path:
```

```
7167                            <1>     ; NOTE: /// 07/05/2016 ///
7168                            <1>     ; If the path is given, value of byte [Run_Manual_Path]
7169                            <1>     ; will not be ZERO. If so, file searching by using
7170                            <1>     ; Automatic Path (via 'PATH' environment variable)
7171                            <1>     ; will not be applicable, because the program file
7172                            <1>     ; is already/absolutely not found.
7173                            <1>
7174 0000A10D A0[FC820100]      <1>     mov     al, [Run_Manual_Path]
7175 0000A112 08C0              <1>     or      al, al
7176                            <1>     ;jnz    loc_cmd_failed
7177                            <1>     ; 25/07/2022
7178 0000A114 752B              <1>     jnz     short loc_run_cap_failed
7179                            <1>
7180                            <1> loc_run_check_auto_path_again:
7181                            <1>     ;cmp    word [Run_Auto_Path], 0FFFFh
7182                            <1>     ;       ; 0FFFFh = Not a valid run path (in ENV block)
7183                            <1>     ;;jnb   loc_cmd_failed
7184                            <1>     ;; 25/07/2022
7185                            <1>     ;jnb    short loc_run_cap_failed
7186                            <1>     ; 03/07/2025
7187 0000A116 FF05[F8820100]    <1>     inc     dword [Run_Auto_Path] ; 0FFFFFFFFh -> 0
7188 0000A11C 7423              <1>     jz      short loc_run_cap_failed ; -1 -> 0
7189                            <1>
7190                            <1>     ; 03/07/2025
7191 0000A11E BF[42780100]      <1>     mov     edi, TextBuffer
7192 0000A123 FF0D[F8820100]    <1>     dec     dword [Run_Auto_Path] ; 0 -> init auto path
7193 0000A129 751D              <1>     jnz     short loc_run_chk_filename_ext_next
7194                            <1>
7195                            <1>     ;xor    al, al
7196 0000A12B BE[322F0100]      <1>     mov     esi, Cmd_Path ; 'PATH'
7197                            <1>     ; 03/07/2025
7198                            <1>     ;mov    edi, TextBuffer
7199                            <1>     ; al = 0
7200                            <1>     ; 03/09/2024 (bugfix)
7201 0000A130 66B90001          <1>     mov     cx, 256 ; TextBuffer (maximum) length
7202 0000A134 E86EF9FFFF        <1>     call    get_environment_string
7203 0000A139 7315              <1>     jnc     short loc_run_chk_filename_ext_again
7204                            <1>     ;mov    word [Run_Auto_Path], 0FFFFh ; invalid
7205                            <1>     ; 03/07/2025
7206 0000A13B FF0D[F8820100]    <1>     dec     dword [Run_Auto_Path] ; -1 (invalid)
7207                            <1> loc_run_cap_failed: ; 25/07/2022
7208                            <1>     ;jmp    loc_cmd_failed
7209                            <1>     ; 30/08/2023
7210 0000A141 B001              <1>     mov     al, 1 ; (force jump to loc_cmd_failed at the end)
7211 0000A143 E927E3FFFF        <1>     jmp     loc_run_cmd_failed ; (restore cdir if it is needed)
7212                            <1>
7213                            <1>     ; 03/07/2025
7214                            <1> loc_run_chk_filename_ext_next:
7215 0000A148 8B0D[F4820100]    <1>     mov     ecx, [Run_Path_Length]
7216 0000A14E EB09              <1>     jmp     short loc_run_chk_filename_ext_again_@
7217                            <1>
7218                            <1> loc_run_chk_filename_ext_again:
7219 0000A150 89C1              <1>     mov     ecx, eax ; string length (with zero tail)
7220 0000A152 49                <1>     dec     ecx ; without zero tail
7221                            <1>     ; 03/07/2025
7222 0000A153 890D[F4820100]    <1>     mov     [Run_Path_Length], ecx
7223                            <1> loc_run_chk_filename_ext_again_@:
7224                            <1>     ; 04/07/2025
7225                            <1>     ;mov    ax, [EXE_ID]
7226                            <1>     ;cmp    ah, '.'
7227                            <1>     ;je     short loc_run_chk_auto_path_pos
7228                            <1>
7229                            <1> loc_run_change_file_ext_to_noext_again:
7230                            <1>     ; 04/07/2025
7231                            <1>     ;movzx  ebx, ah
7232 0000A159 BE[24800100]      <1>     mov     esi, FindFile_Name
7233                            <1>     ;add    ebx, esi
7234 0000A15E 31C0              <1>     xor     eax, eax
7235                            <1> loc_run_change_file_ext_to_noext_@:
7236 0000A160 46                <1>     inc     esi
7237 0000A161 FEC4              <1>     inc     ah
7238                            <1>     ;cmp    al, '.'
7239 0000A163 803E2E            <1>     cmp     byte [esi], '.'
7240 0000A166 75F8              <1>     jne     short loc_run_change_file_ext_to_noext_@
7241 0000A168 66A3[FD820100]    <1>     mov     [EXE_ID], ax
7242                            <1>             ; al = 0, ah = file name length
7243                            <1>     ;sub    eax, eax
7244                            <1>     ;;mov   [ebx], eax ; 0 ; erase extension (.PRG)
7245                            <1>     ;mov    [esi], eax ; 0
7246 0000A16E 8806              <1>     mov     [esi], al ; 0
7247                            <1>
7248                            <1> loc_run_chk_auto_path_pos:
7249                            <1>     ;;movzx eax, word [Run_Auto_Path]
7250                            <1>     ;mov    ax, [Run_Auto_Path]
7251                            <1>     ; 03/07/2025
7252 0000A170 A1[F8820100]      <1>     mov     eax, [Run_Auto_Path]
7253 0000A175 39C8              <1>     cmp     eax, ecx ; ecx = string length (except zero tail)
7254                            <1>     ;jnb    loc_cmd_failed
7255                            <1>     ; 25/07/2022
7256 0000A177 73C8              <1>     jnb     short loc_run_cap_failed
7257                            <1>
7258                            <1>     ; 03/07/2025
7259 0000A179 09C0              <1>     or      eax, eax
7260                            <1>     ;or     ax, ax
7261 0000A17B 7502              <1>     jnz     short loc_run_auto_path_pos_move
7262 0000A17D B005              <1>     mov     al, 5
7263                            <1>
7264                            <1> loc_run_auto_path_pos_move:
7265 0000A17F 89FE              <1>     mov     esi, edi ; offset TextBuffer
7266                            <1>     ; 04/07/2025
7267 0000A181 BF[C2780100]      <1>     mov     edi, RunPathBuffer
7268 0000A186 01C6              <1>     add     esi, eax
7269                            <1>
7270                            <1> loc_run_auto_path_pos_space_loop:
7271 0000A188 AC                <1>     lodsb
7272 0000A189 3C20              <1>     cmp     al, 20h
7273 0000A18B 74FB              <1>     je      short loc_run_auto_path_pos_space_loop
7274                            <1>     ;jb     loc_cmd_failed
7275                            <1>     ; 25/07/2022
7276 0000A18D 72B2              <1>     jb      short loc_run_cap_failed
7277                            <1>     ;
7278                            <1> loc_run_auto_path_pos_move_next_@:
7279 0000A18F AA                <1>     stosb
7280                            <1> loc_run_auto_path_pos_move_next:
7281 0000A190 AC                <1>     lodsb
7282 0000A191 3C3B              <1>     cmp     al, ';'
7283 0000A193 741B              <1>     je      short loc_run_auto_path_pos_move_last_byte
7284 0000A195 3C20              <1>     cmp     al, 20h
7285 0000A197 74F7              <1>     je      short loc_run_auto_path_pos_move_next
7286                            <1>     ;jb     short loc_byte_ptr_end_of_path
7287                            <1>     ;stosb
7288                            <1>     ;jmp    short loc_run_auto_path_pos_move_next
7289                            <1>     ; 03/09/2024
7290 0000A199 73F4              <1>     jnb     short loc_run_auto_path_pos_move_next_@
```

```
7291                                    <1>
7292                                    <1> loc_byte_ptr_end_of_path:
7293 0000A19B 66C705[F8820100]FF-       <1>     mov     word [Run_Auto_Path], 0FFFFh ; end of path
7293 0000A1A3 FF                        <1>
7294                                    <1>     ; 03/07/2025
7295 0000A1A4 C705[F8820100]FFFF-       <1>     mov     dword [Run_Auto_Path], 0FFFFFFFFh
7295 0000A1AC FFFF                      <1>
7296 0000A1AE EB0C                      <1>     jmp     short loc_run_auto_path_move_ok
7297                                    <1>
7298                                    <1> loc_run_auto_path_pos_move_last_byte:
7299 0000A1B0 89F0                      <1>     mov     eax, esi
7300 0000A1B2 2D[42780100]              <1>     sub     eax, TextBuffer
7301                                    <1>     ;mov    [Run_Auto_Path], ax ; next path position
7302                                    <1>     ; 03/07/2025
7303 0000A1B7 A3[F8820100]              <1>     mov     [Run_Auto_Path], eax
7304                                    <1>
7305                                    <1> loc_run_auto_path_move_ok:
7306 0000A1BC 4F                        <1>     dec     edi
7307 0000A1BD B02F                      <1>     mov     al, '/'
7308 0000A1BF 3807                      <1>     cmp     [edi], al
7309 0000A1C1 7403                      <1>     je      short loc_run_auto_path_move_file_name
7310 0000A1C3 47                        <1>     inc     edi
7311 0000A1C4 8807                      <1>     mov     [edi], al
7312                                    <1>
7313                                    <1> loc_run_auto_path_move_file_name:
7314 0000A1C6 47                        <1>     inc     edi
7315 0000A1C7 BE[24800100]              <1>     mov     esi, FindFile_Name
7316                                    <1>
7317                                    <1> loc_run_auto_path_move_fn_loop:
7318 0000A1CC AC                        <1>     lodsb
7319 0000A1CD AA                        <1>     stosb
7320 0000A1CE 08C0                      <1>     or      al, al
7321 0000A1D0 75FA                      <1>     jnz     short loc_run_auto_path_move_fn_loop
7322                                    <1>
7323                                    <1>     ;mov    esi, TextBuffer
7324                                    <1>     ; 04/07/2025
7325 0000A1D2 BE[C2780100]              <1>     mov     esi, RunPathBuffer
7326 0000A1D7 BF[E27F0100]              <1>     mov     edi, FindFile_Drv
7327 0000A1DC E88D070000                <1>     call    parse_path_name
7328                                    <1>     ;jc     loc_run_cmd_failed
7329                                    <1>     ; 25/07/2022
7330                                    <1>     ;jnc    short loc_run_change_current_drive
7331                                    <1>     ;jmp    loc_run_cmd_failed
7332 0000A1E1 723D                      <1>     jc      short loc_run_path_failed
7333                                    <1>
7334                                    <1> loc_run_change_current_drive:
7335 0000A1E3 8A35[42770100]            <1>     mov     dh, [Current_Drv]
7336 0000A1E9 8A15[E27F0100]            <1>     mov     dl, [FindFile_Drv]
7337 0000A1EF 38F2                      <1>     cmp     dl, dh
7338 0000A1F1 7407                      <1>     je      short loc_run_change_directory_again
7339                                    <1>
7340 0000A1F3 E8E8D4FFFF                <1>     call    change_current_drive
7341                                    <1>     ;jc     loc_run_cmd_failed
7342                                    <1>     ; 25/07/2022
7343                                    <1>     ;jnc    short loc_run_change_directory_again
7344                                    <1>     ;jmp    loc_run_cmd_failed
7345 0000A1F8 7226                      <1>     jc      short loc_run_path_failed
7346                                    <1>
7347                                    <1> loc_run_change_directory_again:
7348 0000A1FA 803D[E37F0100]20          <1>     cmp     byte [FindFile_Directory], 20h
7349 0000A201 7627                      <1>     jna     short loc_load_executable_cdir_chk_again
7350                                    <1>
7351 0000A203 FE05[5D2E0100]            <1>     inc     byte [Restore_CDIR]
7352 0000A209 BE[E37F0100]              <1>     mov     esi, FindFile_Directory
7353 0000A20E 30E4                      <1>     xor     ah, ah ; CD_COMMAND sign -> 0
7354 0000A210 E8C6010000                <1>     call    change_current_directory
7355                                    <1>     ;jc     loc_run_cmd_failed
7356                                    <1>     ; 25/07/2022
7357 0000A215 730E                      <1>     jnc     short loc_run_chg_prompt_dir_str_again
7358                                    <1>
7359                                    <1>     ; 03/07/2025
7360                                    <1>     ; ignore 'directory not found' error
7361                                    <1>     ; if auto path in progress
7362 0000A217 803D[FC820100]00          <1>     cmp     byte [Run_Manual_Path], 0
7363 0000A21E 760A                      <1>     jna     short jmp_loc_run_find_executable_file_next
7364                                    <1>
7365                                    <1> loc_run_path_failed: ; 25/07/2022
7366 0000A220 E94AE2FFFF                <1>     jmp     loc_run_cmd_failed
7367                                    <1>
7368                                    <1> loc_run_chg_prompt_dir_str_again:
7369 0000A225 E8DA000000                <1>     call    change_prompt_dir_string
7370                                    <1>
7371                                    <1> loc_load_executable_cdir_chk_again:
7372                                    <1> ; 04/07/2025
7373                                    <1> %if 0
7374                                    <1>     ; 03/07/2025
7375                                    <1>     mov     al, [Current_Drv]
7376                                    <1>     cmp     al, [RUN_CDRV]
7377                                    <1>     jne     short jmp_loc_run_find_executable_file_next
7378                                    <1>     ;
7379                                    <1>     mov     eax, [Current_Dir_FCluster]
7380                                    <1>     cmp     eax, [Run_CDirFC]
7381                                    <1>     ;jne    loc_run_find_executable_file_next
7382                                    <1>     ; 30/08/2023
7383                                    <1>     je      short jmp_loc_run_check_auto_path_again
7384                                    <1>
7385                                    <1> jmp_loc_run_find_executable_file_next: ; 03/07/2025
7386                                    <1>     jmp     loc_run_find_executable_file_next
7387                                    <1> jmp_loc_run_check_auto_path_again:
7388                                    <1>     xor     al, al ; 0
7389                                    <1>     jmp     loc_run_check_auto_path_again
7390                                    <1> %else
7391                                    <1>     ; 04/07/2025
7392                                    <1> jmp_loc_run_find_executable_file_next:
7393 0000A22A E983FEFFFF                <1>     jmp     loc_run_find_executable_file_next
7394                                    <1> %endif
7395                                    <1>
7396                                    <1> loc_load_and_run_file:
7397                                    <1>     ; 25/07/2022 - TRDOS 386 Kernel v2.0.5
7398                                    <1>     ; 13/11/2017
7399                                    <1>     ; 04/01/2017
7400                                    <1>     ; 23/04/2016
7401 0000A22F BE[24800100]              <1>     mov     esi, FindFile_Name
7402 0000A234 BF[42780100]              <1>     mov     edi, TextBuffer
7403                                    <1>
7404                                    <1>     ; 24/04/2016
7405 0000A239 31D2                      <1>     xor     edx, edx
7406                                    <1>     ;mov    word [argc], dx ; 0
7407                                    <1>     ; 25/07/2022
7408 0000A23B 8915[308F0100]            <1>     mov     dword [argc], edx
7409 0000A241 8915[608E0100]            <1>     mov     dword [u.nread], edx ; 0
7410                                    <1>
7411                                    <1> loc_load_and_run_file_1:
7412 0000A247 AC                        <1>     lodsb
```

```
7413 0000A248 AA                <1>     stosb
7414 0000A249 FF05[608E0100]    <1>     inc     dword [u.nread]
7415 0000A24F 20C0              <1>     and     al, al
7416 0000A251 75F4              <1>     jnz     short loc_load_and_run_file_1
7417                            <1>
7418 0000A253 A0[A3770100]      <1>     mov     al, [CmdArgStart]
7419 0000A258 20C0              <1>     and     al, al
7420 0000A25A 7442              <1>     jz      short loc_load_and_run_file_7
7421                            <1>
7422 0000A25C 0FB6F0            <1>     movzx   esi, al ; 11/05/2016
7423                            <1>     ;mov    ecx, 80
7424                            <1>     ; 25/07/2022
7425 0000A25F 31C9              <1>     xor     ecx, ecx
7426 0000A261 B150              <1>     mov     cl, 80
7427                            <1>     ;sub    ecx, esi
7428 0000A263 28C1              <1>     sub     cl, al
7429 0000A265 81C6[F2770100]    <1>     add     esi, CommandBuffer
7430                            <1>
7431                            <1>     ;inc    word [argc] ; 11/05/2016
7432                            <1>     ; 25/07/2022
7433 0000A26B FF05[308F0100]    <1>     inc     dword [argc]
7434                            <1>
7435                            <1> loc_load_and_run_file_2:
7436 0000A271 AC                <1>     lodsb
7437 0000A272 3C20              <1>     cmp     al, 20h
7438 0000A274 7716              <1>     ja      short loc_load_and_run_file_5
7439 0000A276 721D              <1>     jb      short loc_load_and_run_file_6
7440                            <1>
7441                            <1> loc_load_and_run_file_3:
7442 0000A278 803E20            <1>     cmp     byte [esi], 20h
7443 0000A27B 7707              <1>     ja      short loc_load_and_run_file_4
7444 0000A27D 7216              <1>     jb      short loc_load_and_run_file_6
7445 0000A27F 46                <1>     inc     esi
7446 0000A280 E2F6              <1>     loop    loc_load_and_run_file_3
7447 0000A282 EB11              <1>     jmp     short loc_load_and_run_file_6
7448                            <1>
7449                            <1> loc_load_and_run_file_4:
7450 0000A284 28C0              <1>     sub     al, al ; 0
7451                            <1>     ;inc    word [argc]
7452                            <1>     ; 25/07/2022
7453 0000A286 FF05[308F0100]    <1>     inc     dword [argc]
7454                            <1> loc_load_and_run_file_5:
7455 0000A28C AA                <1>     stosb
7456 0000A28D FF05[608E0100]    <1>     inc     dword [u.nread]
7457 0000A293 E2DC              <1>     loop    loc_load_and_run_file_2
7458                            <1>
7459                            <1> loc_load_and_run_file_6:
7460 0000A295 30C0              <1>     xor     al, al ; 0
7461 0000A297 AA                <1>     stosb
7462 0000A298 FF05[608E0100]    <1>     inc     dword [u.nread]
7463                            <1> loc_load_and_run_file_7:
7464 0000A29E 8807              <1>     mov     [edi], al ; 0
7465                            <1>     ;inc    word [argc] ; 24/04/2016
7466                            <1>     ; 25/07/2022
7467 0000A2A0 FF05[308F0100]    <1>     inc     dword [argc]
7468 0000A2A6 FF05[608E0100]    <1>     inc     dword [u.nread] ; 24/04/2016
7469 0000A2AC BE[42780100]      <1>     mov     esi, TextBuffer
7470 0000A2B1 8B15[50800100]    <1>     mov     edx, [FindFile_DirEntry+DirEntry_FileSize]
7471 0000A2B7 66A1[48800100]    <1>     mov     ax, [FindFile_DirEntry+DirEntry_FstClusHI]
7472 0000A2BD C1E010            <1>     shl     eax, 16 ; 13/11/2017
7473 0000A2C0 66A1[4E800100]    <1>     mov     ax, [FindFile_DirEntry+DirEntry_FstClusLO]
7474                            <1>     ; EAX = First Cluster number
7475                            <1>     ; EDX = File Size
7476                            <1>     ; ESI = Argument list address
7477                            <1>     ; [argc] = argument count
7478                            <1>     ; [u.nread] = argument list length
7479 0000A2C6 E8BB620000        <1>     call    load_and_run_file ; trdosk6.s
7480                            <1>        ;jc loc_run_cmd_failed ; 04/01/2017
7481                            <1> loc_load_and_run_file_8: ; 06/05/2016
7482 0000A2CB E968EAFFFF        <1>     jmp     loc_file_rw_restore_retn
7483                            <1>
7484                            <1> check_prg_filename_ext:
7485                            <1>     ; 23/04/2016 (TRDOS 386 = TRDOS v2.0)
7486                            <1>     ; 10/09/2011
7487                            <1>     ; (TRDOS v1, CMDINTR.ASM, 'proc_check_exe_filename_ext')
7488                            <1>     ; 14/11/2009
7489                            <1>     ; INPUT ->
7490                            <1>     ;       ESI = Dot File Name
7491                            <1>     ; OUTPUT ->
7492                            <1>     ;     cf = 0 -> EXE_ID in AL
7493                            <1>     ;       ESI = Last char + 1 position
7494                            <1>     ;     cf = 1 -> Invalid executable file name
7495                            <1>     ;       or no file name extension if AH<=8
7496                            <1>     ;       AL = Last file name char
7497                            <1>     ;     cf = 0 -> AL='P' (PRG), AL=0 (no extension)
7498                            <1>     ;
7499                            <1>     ; (Modified registers: EAX, ESI)
7500                            <1>
7501 0000A2D0 30E4              <1>     xor     ah, ah
7502                            <1> loc_run_check_filename_ext:
7503 0000A2D2 AC                <1>     lodsb
7504 0000A2D3 3C21              <1>     cmp     al, 21h
7505 0000A2D5 7229              <1>     jb      short loc_check_exe_fn_retn
7506 0000A2D7 FEC4              <1>     inc     ah
7507 0000A2D9 3C2E              <1>     cmp     al, '.'
7508 0000A2DB 75F5              <1>     jne     short loc_run_check_filename_ext
7509                            <1>
7510                            <1> loc_run_check_filename_ext_dot:
7511 0000A2DD 80FC02            <1>     cmp     ah, 2  ; .??? is not valid
7512 0000A2E0 88C4              <1>     mov     ah, al ; '.'
7513 0000A2E2 7219              <1>     jb      short loc_check_prg_fn_retn
7514                            <1>
7515                            <1> loc_run_check_filename_ext_dot_ok:
7516 0000A2E4 AC                <1>     lodsb
7517 0000A2E5 24DF              <1>     and     al, 0DFh
7518                            <1>
7519                            <1> loc_run_check_filename_ext_prg:
7520 0000A2E7 3C50              <1>     cmp     al, 'P'
7521 0000A2E9 7212              <1>     jb      short loc_check_prg_fn_retn
7522 0000A2EB 7711              <1>     ja      short loc_check_prg_fn_stc
7523 0000A2ED AC                <1>     lodsb
7524 0000A2EE 24DF              <1>     and     al, 0DFh
7525 0000A2F0 3C52              <1>     cmp     al, 'R'
7526 0000A2F2 750A              <1>     jne     short loc_check_prg_fn_stc
7527 0000A2F4 AC                <1>     lodsb
7528 0000A2F5 24DF              <1>     and     al, 0DFh
7529 0000A2F7 3C47              <1>     cmp     al, 'G'
7530 0000A2F9 7503              <1>     jne     short loc_check_prg_fn_stc
7531                            <1>
7532 0000A2FB B050              <1>     mov     al, 'P'
7533                            <1> loc_check_prg_fn_retn:
7534 0000A2FD C3                <1>     retn
7535                            <1>
7536                            <1> ; 25/07/2022 - TRDOS 386 Kernel v2.0.5
```

```
7537                                    <1>
7538                                    <1> loc_check_prg_fn_stc:
7539 0000A2FE F9                       <1>     stc
7540 0000A2FF C3                       <1>     retn
7541                                    <1>
7542                                    <1> loc_check_exe_fn_retn:
7543 0000A300 28C0                     <1>     sub     al, al ; 0
7544                                    <1>     ;retn
7545                                    <1>
7546                                    <1> find_and_list_files:
7547                                    <1>     ;retn
7548                                    <1> set_exec_arguments:
7549 0000A302 C3                       <1>     retn
7550                                    <1>
7551                                    <1> ; 19/12/2025
7552                                    <1> ;delete_fs_directory:
7553                                    <1> ;   xor     eax, eax
7554 0000A303 C3                       <1>     retn
3435                                        %include 'trdosk4.s' ; 24/01/2016
   1                                    <1> ; ****************************************************************************
   2                                    <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.10) - Directory Functions : trdosk4.s
   3                                    <1> ; ----------------------------------------------------------------------------
   4                                    <1> ; Last Update: 19/12/2025  (Previous: 03/09/2024, v2.0.9)
   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 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 0000A304 BE[A07E0100]              <1>     mov     esi, PATH_Array
  28                                    <1> change_prompt_dir_str: ; 05/10/2016 (call from 'set_working_path')
  29 0000A309 BF[46770100]              <1>     mov     edi, Current_Directory
  30 0000A30E 8A25[40770100]            <1>     mov     ah, [Current_Dir_Level]
  31 0000A314 E807000000                <1>     call    set_current_directory_string
  32 0000A319 880D[A1770100]            <1>     mov     [Current_Dir_StrLen], cl
  33                                    <1>
  34 0000A31F C3                        <1>     retn
  35                                    <1>
  36                                    <1> set_current_directory_string:
  37                                    <1>     ; 11/08/2022 (TRDOS 386 Kernel v2.0.5)
  38                                    <1>     ; 24/01/2016 (TRDOS 386 = TRDOS v2.0)
  39                                    <1>     ; 27/03/2011
  40                                    <1>     ; 09/10/2009
  41                                    <1>     ; INPUT:
  42                                    <1>     ;    ESI = Path Array Address
  43                                    <1>     ;    EDI = Current Directory String Buffer
  44                                    <1>     ;    AH = Current Directory Level
  45                                    <1>     ; OUTPUT => EAX, EBX, ESI will be changed
  46                                    <1>     ;    EDI will be same with input
  47                                    <1>     ;    ECX = Current Directory String Length
  48                                    <1>
  49 0000A320 57                        <1>     push    edi
  50 0000A321 80FC00                    <1>     cmp     ah, 0
  51 0000A324 7651                      <1>     jna     short pass_write_path
  52 0000A326 83C610                    <1>     add     esi, 16
  53 0000A329 89F3                      <1>     mov     ebx, esi
  54                                    <1> loc_write_path:
  55                                    <1>     ;mov    ecx, 8
  56                                    <1>     ; 11/08/2022
  57 0000A32B 29C9                      <1>     sub     ecx, ecx
  58 0000A32D B108                      <1>     mov     cl, 8
  59                                    <1> path_write_dirname1:
  60 0000A32F AC                        <1>     lodsb
  61 0000A330 3C20                      <1>     cmp     al, 20h
  62 0000A332 7612                      <1>     jna     short pass_write_dirname1
  63 0000A334 AA                        <1>     stosb
  64 0000A335 81FF[A0770100]            <1>     cmp     edi, End_Of_Current_Dir_Str
  65 0000A33B 733A                      <1>     jnb     short pass_write_path
  66 0000A33D E2F0                      <1>     loop    path_write_dirname1
  67 0000A33F 803E20                    <1>     cmp     byte [esi], 20h
  68 0000A342 7624                      <1>     jna     short pass_write_dirname2
  69 0000A344 EB0A                      <1>     jmp     short loc_put_dot_cont_ext
  70                                    <1> pass_write_dirname1:
  71 0000A346 89DE                      <1>     mov     esi, ebx
  72 0000A348 83C608                    <1>     add     esi, 8
  73 0000A34B 803E20                    <1>     cmp     byte [esi], 20h
  74 0000A34E 7618                      <1>     jna     short pass_write_dirname2
  75                                    <1> loc_put_dot_cont_ext:
  76 0000A350 C6072E                    <1>     mov     byte [edi], "."
  77                                    <1>     ;mov    ecx, 3
  78 0000A353 B103                      <1>     mov     cl, 3
  79                                    <1> loc_check_dir_name_ext:
  80 0000A355 AC                        <1>     lodsb
  81 0000A356 47                        <1>     inc     edi
  82 0000A357 3C20                      <1>     cmp     al, 20h
  83 0000A359 760D                      <1>     jna     short pass_write_dirname2
  84 0000A35B 8807                      <1>     mov     [edi], al
  85 0000A35D 81FF[A0770100]            <1>     cmp     edi, End_Of_Current_Dir_Str
  86 0000A363 7312                      <1>     jnb     short pass_write_path
  87 0000A365 E2EE                      <1>     loop    loc_check_dir_name_ext
  88 0000A367 47                        <1>     inc     edi
  89                                    <1> pass_write_dirname2:
  90 0000A368 FECC                      <1>     dec     ah
  91 0000A36A 740B                      <1>     jz      short pass_write_path
  92 0000A36C 83C310                    <1>     add     ebx, 16
  93 0000A36F 89DE                      <1>     mov     esi, ebx
  94 0000A371 C6072F                    <1>     mov     byte [edi],"/"
  95 0000A374 47                        <1>     inc     edi
  96 0000A375 EBB4                      <1>     jmp     short loc_write_path
  97                                    <1> pass_write_path:
  98 0000A377 C60700                    <1>     mov     byte [edi], 0
  99 0000A37A 47                        <1>     inc     edi
 100 0000A37B 89F9                      <1>     mov     ecx, edi
 101 0000A37D 5F                        <1>     pop     edi
 102 0000A37E 29F9                      <1>     sub     ecx, edi
 103                                    <1>     ; ECX = Current Directory String Length
 104 0000A380 C3                        <1>     retn
 105                                    <1>
```

```
                                <1> get_current_directory:
                                <1>     ; 29/08/2023 (TRDOS 386 Kernel v2.0.6)
                                <1>     ; 11/08/2022
                                <1>     ; 28/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <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:
                                <1>     ; 29/08/2023
0000A381 29C0                   <1>     sub     eax, eax ; 0
                                <1>     ;cmp    dl, 0
0000A383 38C2                   <1>     cmp     dl, al
0000A385 7724                   <1>     ja      short loc_get_current_drive_1
0000A387 8A15[42770100]         <1>     mov     dl, [Current_Drv]
                                <1>     ; 29/08/2023
                                <1>     ;jmp    short loc_get_current_drive_2
                                <1>
                                <1> loc_get_current_drive_2:
                                <1>     ; 29/08/2023
                                <1>     ;xor    eax, eax
                                <1>     ; eax = 0
0000A38D 88D4                   <1>     mov     ah, dl
0000A38F 56                     <1>     push    esi ; (*)
0000A390 BE00010900             <1>     mov     esi, Logical_DOSDisks
0000A395 01C6                   <1>     add     esi, eax
0000A397 8A06                   <1>     mov     al, [esi+LD_Name]
0000A399 3C41                   <1>     cmp     al, 'A'
0000A39B 721C                   <1>     jb      short loc_get_current_drive_not_ready_retn
                                <1>
                                <1>     ; 28/07/2022
0000A39D 31C9                   <1>     xor     ecx, ecx
                                <1>
0000A39F 8A667F                 <1>     mov     ah, [esi+LD_CDirLevel]
0000A3A2 08E4                   <1>     or      ah, ah
0000A3A4 7519                   <1>     jnz     short loc_get_current_drive_3
                                <1>
                                <1>     ;xor    ah, ah ; mov ah, 0
                                <1>
                                <1>     ; 11/08/2022 - BugFix (*)
0000A3A6 5E                     <1>     pop     esi ; (*) Current Directory Buffer address
                                <1>
0000A3A7 8826                   <1>     mov     [esi], ah
                                <1>     ; 28/07/2022
                                <1>     ;xor    ecx, ecx
0000A3A9 EB2D                   <1>     jmp     short loc_get_current_drive_4
                                <1>
                                <1>     ; 29/08/2023
                                <1> loc_get_current_drive_1:
0000A3AB FECA                   <1>     dec     dl
0000A3AD 3A15[5C2E0100]         <1>     cmp     dl, [Last_DOS_DiskNo]
0000A3B3 76D8                   <1>     jna     short loc_get_current_drive_2
                                <1>     ;mov    eax, 0Fh ; Invalid drive (Drive not ready!)
                                <1>     ;cmc    ; stc
                                <1>     ; 28/07/2022
                                <1>     ;sub    eax, eax ; 29/08/2023
                                <1>     ; eax = 0
0000A3B5 B00F                   <1>     mov     al, 0Fh
0000A3B7 F9                     <1>     stc
0000A3B8 C3                     <1>     retn
                                <1>
                                <1> loc_get_current_drive_not_ready_retn:
0000A3B9 5E                     <1>     pop     esi
                                <1>     ;mov    eax, 15
0000A3BA 66B80F00               <1>     mov     ax, 15 ; Drive not ready
0000A3BE C3                     <1>     retn
                                <1>
                                <1> loc_get_current_drive_3:
0000A3BF BF[A07E0100]           <1>         mov     edi, PATH_Array
0000A3C4 57                     <1>     push    edi
0000A3C5 81C680000000           <1>     add     esi, LD_CurrentDirectory
                                <1>     ;mov    ecx, 32
                                <1>     ; 28/07/2022
0000A3CB B120                   <1>     mov     cl, 32
0000A3CD F3A5                   <1>     rep     movsd
0000A3CF 5E                     <1>     pop     esi ; Path Array Address
0000A3D0 5F                     <1>     pop     edi ; pushed esi (current dir buffer offset)
                                <1>     ;
0000A3D1 E84AFFFFFF             <1>     call    set_current_directory_string
0000A3D6 89FE                   <1>     mov     esi, edi
                                <1>
                                <1> loc_get_current_drive_4:
0000A3D8 30C0                   <1>     xor     al, al
0000A3DA C3                     <1>     retn
                                <1>
                                <1> change_current_directory:
                                <1>     ; 19/12/2025 (TRDOS 386 Kernel v2.0.10)
                                <1>     ; 28/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <1>     ; 02/03/2021 (TRDOS 386 v2.0.3) ((BugFix))
                                <1>     ; 19/02/2016
                                <1>     ; 11/02/2016
                                <1>     ; 10/02/2016
                                <1>     ; 08/02/2016
                                <1>     ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>     ; 18/09/2011 (DIR.ASM, 09/10/2011)
                                <1>     ; 04/10/2009
                                <1>     ; 2005
                                <1>     ; INPUT ->
                                <1>     ;      ESI = Directory string
                                <1>     ;      ah = CD command (CDh = save current dir string)
                                <1>     ; OUTPUT ->
                                <1>     ;      EDI = DOS Drive Description Table
                                <1>     ;      cf = 1 -> error
                                <1>     ;         EAX = Error code
                                <1>     ;      cf = 0 -> successful
                                <1>     ;         ESI = PATH_Array
                                <1>     ;         EAX = Current Directory First Cluster
                                <1>     ;
```

```
                            <1>     ; (EAX, EBX, ECX, EDX, ESI, EDI will be changed)
                            <1>
0000A3DB 8825[2E7F0100]     <1>     mov     [CD_COMMAND], ah
0000A3E1 803E2F             <1>     cmp     byte [esi], '/'
0000A3E4 7505               <1>     jne     short loc_ccd_cdir_level
0000A3E6 46                 <1>     inc     esi
0000A3E7 30C0               <1>     xor     al, al
0000A3E9 EB05               <1>     jmp     short loc_ccd_parse_path_name
                            <1> loc_ccd_cdir_level:
0000A3EB A0[40770100]       <1>     mov     al, [Current_Dir_Level]
                            <1> loc_ccd_parse_path_name:
0000A3F0 88C4               <1>     mov     ah, al
0000A3F2 BF[A07E0100]       <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>
0000A3F7 0FB6C8             <1>     movzx   ecx, al
0000A3FA FEC1               <1>     inc     cl
0000A3FC C0E104             <1>     shl     cl, 4
0000A3FF 01CF               <1>     add     edi, ecx
0000A401 B107               <1>     mov     cl, 7
0000A403 28C1               <1>     sub     cl, al
0000A405 C0E102             <1>     shl     cl, 2
0000A408 89C3               <1>     mov     ebx, eax
0000A40A 31C0               <1>     xor     eax, eax ; 0
0000A40C F3AB               <1>     rep     stosd
0000A40E 89D8               <1>     mov     eax, ebx
                            <1>
0000A410 BF[A07E0100]       <1>     mov     edi, PATH_Array
                            <1>
0000A415 803E20             <1>     cmp     byte [esi], 20h
0000A418 F5                 <1>     cmc
0000A419 7305               <1>     jnc     short pass_ccd_parse_dir_name
                            <1>
                            <1>             ; ESI = Path name
                            <1>             ; AL = CCD_Level
0000A41B E83B010000         <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:
0000A420 9C                 <1>     pushf
                            <1>
                            <1>     ;mov    [CCD_Level], al
                            <1>         ;mov        [Last_Dir_Level], ah
0000A421 66A3[247F0100]     <1>     mov     [CCD_Level], ax
                            <1>
0000A427 31DB               <1>     xor     ebx, ebx
0000A429 8A3D[42770100]     <1>     mov     bh, [Current_Drv]
0000A42F BE00010900         <1>     mov     esi, Logical_DOSDisks
0000A434 01DE               <1>     add     esi, ebx
                            <1>
0000A436 9D                 <1>     popf
0000A437 720A               <1>     jc      short loc_ccd_bad_path_name_retn
                            <1>
0000A439 8935[207F0100]     <1>     mov     [CCD_DriveDT], esi
                            <1>
0000A43F 3C07               <1>     cmp     al, 7
0000A441 7208               <1>     jb      short loc_ccd_load_child_dir
                            <1>
                            <1> loc_ccd_bad_path_name_retn:
0000A443 87FE               <1>     xchg    esi, edi
                            <1>     ;mov    eax, 19 ; Bad directory/path name
                            <1>     ; 28/07/2022
0000A445 29C0               <1>     sub     eax, eax
0000A447 B013               <1>     mov     al, 19
0000A449 F9                 <1>     stc
                            <1> loc_ccd_retn_p:
0000A44A C3                 <1>     retn
                            <1>
                            <1> loc_ccd_load_child_dir:
                            <1>     ; AL = CCD_Level
0000A44B 08C0               <1>     or      al, al
0000A44D 744C               <1>     jz      short loc_ccd_load_root_dir
                            <1>
                            <1>     ;mov    cx, ax
                            <1>     ; 28/07/2022
0000A44F 89C1               <1>     mov     ecx, eax
0000A451 C0E004             <1>     shl     al, 4
0000A454 0FB6F0             <1>     movzx   esi, al
0000A457 01FE               <1>             add     esi, edi  ; offset PATH_Array
                            <1>
0000A459 8B460C             <1>     mov     eax, [esi+12]
0000A45C 38E9               <1>     cmp     cl, ch
                            <1>     ;je     loc_ccd_load_sub_directory
                            <1>     ; 28/07/2022
0000A45E 7505               <1>     jne     short loc_ccd_1
0000A460 E992000000         <1>     jmp     loc_ccd_load_sub_directory
                            <1>
                            <1> loc_ccd_1:  ; 28/07/2022
0000A465 A3[3C770100]       <1>     mov     [Current_Dir_FCluster], eax
                            <1>
                            <1> loc_ccd_load_child_dir_next:
0000A46A 83C610             <1>     add     esi, 16 ; DOS DirEntry Format FileName Address
                            <1>
                            <1>     ; Directory attribute : 10h
0000A46D B010               <1>     mov     al, 00010000b ; 10h (Attrib AND mask)
                            <1>     ;mov    ah, 11001000b ; C8h
                            <1>     ; Volume name attribute: 8h
0000A46F B408               <1>     mov     ah, 00001000b ; 08h (Attrib NAND, AND --> zero mask)
                            <1>
                            <1>     ;xor    cx, cx
0000A471 31C9               <1>     xor     ecx, ecx ; 02/03/2021
0000A473 E880010000         <1>     call    locate_current_dir_file
0000A478 733C               <1>     jnc     short loc_ccd_set_dir_cluster_ptr
                            <1>
                            <1>     ; 19/02/2016
                            <1>     ;mov    edi, [CCD_DriveDT]
0000A47A 8A25[247F0100]     <1>     mov     ah, [CCD_Level]
0000A480 803D[2E7F0100]CD   <1>     cmp     byte [CD_COMMAND], 0CDh ;'CD' command or another
0000A487 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.)
0000A489 88E1               <1>     mov     cl, ah
```

```
0000A48B 50                  <1>     push    eax
0000A48C E8AE000000          <1>     call    loc_ccd_save_current_dir
0000A491 58                  <1>     pop     eax
                             <1> loc_ccd_load_child_dir_err:
0000A492 3C03                <1>     cmp     al, 3  ; AL = 2 => File not found error
0000A494 7202                <1>     jb      short loc_ccd_path_not_found_retn
0000A496 F9                  <1>     stc
0000A497 C3                  <1>     retn
                             <1>
                             <1> loc_ccd_path_not_found_retn:
0000A498 B003                <1>     mov     al, 3  ; Path not found
0000A49A C3                  <1>     retn
                             <1>
                             <1> loc_ccd_load_root_dir:     ; 19/12/2025
                             <1> loc_ccd_load_FAT_root_dir:
0000A49B 803D[41770100]02    <1>     cmp     byte [Current_FATType], 2
0000A4A2 774B                <1>     ja      short loc_ccd_load_FAT32_root_dir
                             <1>
                             <1>     ;mov    esi, [CCD_DriveDT]
                             <1>     ;push   esi
0000A4A4 E80E1C0000          <1>     call    load_FAT_root_directory
                             <1>     ;pop    edi ; Dos Drv Description Table
                             <1>
0000A4A9 89F7                <1>     mov     edi, esi
0000A4AB BE[A07E0100]        <1>     mov     esi, PATH_Array
0000A4B0 7298                <1>     jc      short loc_ccd_retn_p
                             <1>
0000A4B2 31C0                <1>     xor     eax, eax
0000A4B4 EB58                <1>         jmp short loc_ccd_set_cdfc
                             <1>
                             <1> ; 19/12/2025
                             <1> %if 0
                             <1> loc_ccd_load_root_dir:
                             <1>     cmp     byte [Current_FATType], 1
                             <1>     jnb     short loc_ccd_load_FAT_root_dir
                             <1>
                             <1> loc_ccd_load_FS_root_dir:
                             <1>     call    load_FS_root_directory
                             <1>     jmp     short pass_ccd_load_FAT_sub_directory
                             <1>
                             <1> loc_ccd_load_FS_sub_directory_next:
                             <1>     call    load_FS_sub_directory
                             <1>     jmp     short pass_ccd_set_dir_cluster_ptr
                             <1> %endif
                             <1>
                             <1> loc_ccd_set_dir_cluster_ptr:
                             <1>     ; EDI = Directory Entry
0000A4B6 668B4714            <1>     mov     ax, [edi+20] ; First Cluster High Word
0000A4BA C1E010              <1>     shl     eax, 16
0000A4BD 668B471A            <1>     mov     ax, [edi+26] ; First Cluster Low Word
                             <1>
0000A4C1 8B35[207F0100]      <1>     mov     esi, [CCD_DriveDT]
                             <1>
                             <1> ; 19/12/2025
                             <1> %if 0
                             <1>     cmp     byte [Current_FATType], 1
                             <1>     jb      short loc_ccd_load_FS_sub_directory_next
                             <1> %endif
                             <1>     ;push   esi
0000A4C7 E8691C0000          <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
0000A4CC BE[A07E0100]        <1>     mov     esi, PATH_Array
0000A4D1 7265                <1>     jc      short loc_ccd_retn_c
                             <1>
0000A4D3 A1[6E7E0100]        <1>     mov     eax, [DirBuff_Cluster]
                             <1>
0000A4D8 FE05[247F0100]      <1>     inc     byte [CCD_Level]
0000A4DE 0FB61D[247F0100]    <1>     movzx   ebx, byte [CCD_Level]
0000A4E5 C0E304              <1>     shl     bl, 4 ; * 16 (<= 128)
0000A4E8 01DE                <1>     add     esi, ebx ; 19/02/2016
0000A4EA 89460C              <1>     mov     [esi+12], eax
0000A4ED EB1F                <1>     jmp     short loc_ccd_set_cdfc
                             <1>
                             <1> loc_ccd_load_FAT32_root_dir:
0000A4EF BE[A07E0100]        <1>     mov     esi, PATH_Array
0000A4F4 8B460C              <1>     mov     eax, [esi+12]
                             <1>
                             <1> loc_ccd_load_sub_directory: ; 19/12/2025
0000A4F7 8B35[207F0100]      <1>     mov     esi, [CCD_DriveDT]
                             <1>
                             <1> loc_ccd_load_FAT_sub_directory:
                             <1>     ;push   esi
0000A4FD E8331C0000          <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
0000A502 BE[A07E0100]        <1>     mov     esi, PATH_Array
0000A507 722F                <1>     jc      short loc_ccd_retn_c
                             <1>
0000A509 A1[6E7E0100]        <1>     mov     eax, [DirBuff_Cluster]
                             <1>
                             <1> loc_ccd_set_cdfc:
0000A50E 8A0D[247F0100]      <1>     mov     cl, [CCD_Level]
0000A514 880D[40770100]      <1>     mov     [Current_Dir_Level], cl
0000A51A A3[3C770100]        <1>     mov     [Current_Dir_FCluster], eax
                             <1>
0000A51F 8A2D[257F0100]      <1>     mov     ch, [Last_Dir_Level]
0000A525 38E9                <1>     cmp     cl, ch
                             <1>     ;jb     loc_ccd_load_child_dir_next
                             <1>     ; 28/07/2022
0000A527 7305                <1>     jnb     short loc_ccd_2
0000A529 E93CFFFFFF          <1>     jmp     loc_ccd_load_child_dir_next
                             <1> loc_ccd_2:
0000A52E 803D[2E7F0100]CD    <1>     cmp     byte [CD_COMMAND], 0CDh ;'CD' command or another
0000A535 7408                <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>
0000A537 F8                  <1>     clc
                             <1>
                             <1> loc_ccd_retn_c:
0000A538 8B3D[207F0100]      <1>     mov     edi, [CCD_DriveDT]
0000A53E C3                  <1>     retn
                             <1>
                             <1> ; 19/12/2025
                             <1> %if 0
                             <1> loc_ccd_load_sub_directory:
                             <1>     mov     esi, [CCD_DriveDT]
                             <1>     cmp     byte [Current_FATType], 1
```

```
                                <1>     jnb     short loc_ccd_load_FAT_sub_directory
                                <1>     call    load_FS_sub_directory
                                <1>     jmp     short pass_ccd_load_FAT_sub_directory
                                <1> %endif
                                <1>
                                <1> loc_ccd_save_current_dir:
                                <1>     ; 02/03/2021 (TRDOS 386 v2.0.3) ((BugFix))
                                <1>     ; ('find_directory_entry' has been fixed to prevent large
                                <1>     ; ECX value > 65535)
                                <1>     ;
0000A53F BE[A07E0100]           <1>     mov     esi, PATH_Array ; 19/02/2016
0000A544 8B3D[207F0100]         <1>     mov     edi, [CCD_DriveDT]
0000A54A 57                     <1>     push    edi
0000A54B 83C77F                 <1>         add     edi, LD_CDirLevel
0000A54E 880F                   <1>     mov     [edi], cl
0000A550 47                     <1>     inc     edi ; LD_CurrentDirectory
0000A551 56                     <1>     push    esi
                                <1>     ;;mov   ecx, 32  ; always < 65536 (in this procedure)
0000A552 66B92000               <1>     mov     cx, 32
                                <1>     ; 02/03/2021
                                <1>     ;mov    ecx, 32
0000A556 F3A5                   <1>     rep     movsd
                                <1>     ; Current directory has been saved to
                                <1>     ; the DOS drive description table, cdir area !
0000A558 5E                     <1>     pop     esi  ; PATH_Array
0000A559 5F                     <1>     pop     edi  ; Dos Drv Description Table
                                <1>
0000A55A 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
0000A55B 88C4                   <1>     mov     ah, al
0000A55D 66A3[C47F0100]         <1>     mov     [PATH_CDLevel], ax
                                <1> repeat_ppdn_check_slash:
0000A563 AC                     <1>     lodsb
0000A564 3C2F                   <1>     cmp     al, '/'
0000A566 74FB                   <1>     je      short repeat_ppdn_check_slash
0000A568 3C21                   <1>     cmp     al, 21h
0000A56A 7219                   <1>     jb      short loc_ppdn_retn
0000A56C 57                     <1>     push    edi
                                <1> loc_ppdn_get_dir_name:
0000A56D B90C000000             <1>     mov     ecx, 12
0000A572 BF[C67F0100]           <1>     mov     edi, Dir_File_Name
                                <1> repeat_ppdn_get_dir_name:
0000A577 AA                     <1>     stosb
0000A578 AC                     <1>     lodsb
0000A579 3C2F                   <1>     cmp     al, '/'
0000A57B 740A                   <1>     je      short loc_check_level_dot_conv_dir_name
0000A57D 3C20                   <1>     cmp     al, 20h
0000A57F 7605                   <1>     jna     short loc_ppdn_end_of_path_scan
0000A581 E2F4                   <1>     loop    repeat_ppdn_get_dir_name
0000A583 5F                     <1>     pop     edi
0000A584 F9                     <1>     stc
                                <1> loc_ppdn_retn:
0000A585 C3                     <1>     retn
                                <1>
                                <1> loc_ppdn_end_of_path_scan:
0000A586 4E                     <1>     dec     esi
                                <1> loc_check_level_dot_conv_dir_name:
0000A587 31C0                   <1>     xor     eax, eax
0000A589 AA                     <1>     stosb
0000A58A 89F3                   <1>     mov     ebx, esi
0000A58C BE[C67F0100]           <1>     mov     esi, Dir_File_Name
0000A591 AC                     <1>     lodsb
                                <1> repeat_ppdn_name_check_dot:
0000A592 3C2E                   <1>     cmp     al, '.'
0000A594 7509                   <1>     jne     short loc_ppdn_convert_sub_dir_name
                                <1> repeat_ppdn_name_dot_dot:
0000A596 AC                     <1>     lodsb
0000A597 3C2E                   <1>     cmp     al, '.'
0000A599 743E                   <1>     je      short loc_ppdn_dot_dot
0000A59B 3C21                   <1>     cmp     al, 21h
0000A59D 7226                   <1>     jb      short pass_ppdn_convert_sub_dir_name
                                <1> loc_ppdn_convert_sub_dir_name:
0000A59F 8A25[C57F0100]         <1>     mov     ah, [PATH_Level]
0000A5A5 80FC07                 <1>     cmp     ah, 7
0000A5A8 731B                   <1>     jnb     short pass_ppdn_convert_sub_dir_name
0000A5AA FEC4                   <1>     inc     ah
0000A5AC 8825[C57F0100]         <1>     mov     [PATH_Level], ah
0000A5B2 BE[C67F0100]           <1>     mov     esi, Dir_File_Name
                                <1>     ;mov    edi, [PATH_Array_Ptr]
0000A5B7 B010                   <1>     mov     al, 16
0000A5B9 F6E4                   <1>     mul     ah
0000A5BB 8B3C24                 <1>     mov     edi, [esp]
                                <1>     ;push   edi
0000A5BE 01C7                   <1>     add     edi, eax
0000A5C0 E8DF020000             <1>     call    convert_file_name
                                <1>     ;pop    edi
                                <1> pass_ppdn_convert_sub_dir_name:
0000A5C5 89DE                   <1>     mov     esi, ebx
                                <1> repeat_ppdn_check_last_slash:
0000A5C7 AC                     <1>     lodsb
0000A5C8 3C2F                   <1>     cmp     al, '/'
0000A5CA 74FB                   <1>     je      short repeat_ppdn_check_last_slash
0000A5CC 3C21                   <1>     cmp     al, 21h
0000A5CE 739D                   <1>     jnb     short loc_ppdn_get_dir_name
                                <1> end_of_parse_dir_name:
0000A5D0 5F                     <1>     pop     edi
0000A5D1 F5                     <1>     cmc
                                <1>     ;mov    al, [PATH_CDLevel]
                                <1>     ;mov    ah, [PATH_Level]
0000A5D2 66A1[C47F0100]         <1>     mov     ax, [PATH_CDLevel]
0000A5D8 C3                     <1>     retn
                                <1>
                                <1> loc_ppdn_dot_dot:
0000A5D9 AC                     <1>     lodsb
```

```
602 0000A5DA 3C21                <1>     cmp     al, 21h
603 0000A5DC 73F2                <1>     jnb     short end_of_parse_dir_name
604                              <1> loc_ppdn_dot_dot_prev_level:
605 0000A5DE 66A1[C47F0100]      <1>     mov     ax, [PATH_CDLevel]
606 0000A5E4 80EC01              <1>     sub     ah, 1
607 0000A5E7 80D400              <1>     adc     ah, 0
608 0000A5EA 38E0                <1>     cmp     al, ah
609 0000A5EC 7602                <1>     jna     short pass_ppdn_set_al_to_ah
610 0000A5EE 88E0                <1>     mov     al, ah
611                              <1> pass_ppdn_set_al_to_ah:
612 0000A5F0 66A3[C47F0100]      <1>     mov     [PATH_CDLevel], ax
613 0000A5F6 EBCD                <1>     jmp     short pass_ppdn_convert_sub_dir_name
614                              <1>
615                              <1> locate_current_dir_file:
616                              <1>     ; 19/12/2025 (TRDOS 386 Kernel v2.0.10)
617                              <1>     ; 26/08/2024 (TRDOS 386 v2.0.9)
618                              <1>     ; 26/08/2024 (TRDOS 386 v2.0.9)
619                              <1>     ; 28/07/2022 (TRDOS 386 Kernel v2.0.5)
620                              <1>     ; 20/11/2017
621                              <1>     ; 14/02/2016
622                              <1>     ; 13/02/2016
623                              <1>     ; 10/02/2016
624                              <1>     ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
625                              <1>     ; 14/08/2010
626                              <1>     ; 19/09/2009
627                              <1>       ; 2005
628                              <1>     ; INPUT ->
629                              <1>     ;       ESI = DOS DirEntry Format FileName Address
630                              <1>     ;       AL = Attributes Mask
631                              <1>     ;       (<AL AND EntryAttrib> must be equal to AL)
632                              <1>     ;       AH = Negative Attributes Mask (If AH>0)
633                              <1>     ;       (<AH AND EntryAttrib> must be ZERO)
634                              <1>     ;       CH > 0 Find First Free Dir Entry or Deleted Entry
635                              <1>     ;       CL = 0 -> Return the First Free Dir Entry
636                              <1>     ;       CL = E5h -> Return the 1st deleted entry
637                              <1>     ;       CL = FFh -> Return the 1st deleted or free entry
638                              <1>     ;       CL > 0 and CL <> E5h and CL <> FFh -> Return the first
639                              <1>     ;           proper entry (which fits with Atributes Masks)
640                              <1>     ;       CX = 0 Find Valid File/Directory/VolumeName
641                              <1>     ;       ? = Any One Char
642                              <1>     ;       * = Every Chars
643                              <1>     ; OUTPUT ->
644                              <1>     ;       EDI = Directory Entry Address (in Directory Buffer)
645                              <1>     ;       ESI = DOS DirEntry Format FileName Address
646                              <1>     ;       CF = 0 -> No Error, Proper Entry,
647                              <1>     ;       DL = Attributes
648                              <1>     ;       DH = Previous Entry Attr (LongName Check)
649                              <1>     ;       AL > 0 -> Ambiguous filename wildcard "?" used
650                              <1>     ;       AH > 0 -> Ambiguous filename wildcard "*" used
651                              <1>     ;       AX = 0 -> Filename full fits with directory entry
652                              <1>     ;       CH = The 1st Name Char of Current Dir Entry
653                              <1>     ;       CF = 1 -> Proper entry not found, Error Code in EAX/AL
654                              <1>     ;       CL = 0 and CH = 0 -> Free Entry (End Of Dir)
655                              <1>     ;       CL = 0 and CH = E5h -> Deleted Entry fits with filters
656                              <1>     ;       CL > 0 -> Entry not found, CH invalid
657                              <1>     ;       CF = 0 ->
658                              <1>     ;       EBX = Current Directory Entry Index/Number (BX)
659                              <1>
660                              <1>     ;mov    word [DirBuff_EntryCounter], 0 ; Zero Based
661                              <1>
662 0000A5F8 8935[287F0100]      <1>     mov     [CDLF_FNAddress], esi
663 0000A5FE 66A3[267F0100]      <1>     mov     [CDLF_AttributesMask], ax
664 0000A604 66890D[2C7F0100]    <1>     mov     [CDLF_DEType], cx
665                              <1>
666 0000A60B 31DB                <1>     xor     ebx, ebx
667 0000A60D 881D[3C7F0100]      <1>     mov     [PreviousAttr], bl ; 0  ; 13/02/2016
668                              <1>
669 0000A613 8A3D[42770100]      <1>     mov     bh, [Current_Drv]
670 0000A619 381D[697E0100]      <1>     cmp     byte [DirBuff_ValidData], bl ; 0
671 0000A61F 761D                <1>     jna     short loc_lcdf_reload_current_dir2
672 0000A621 8A1D[677E0100]      <1>         mov     bl, [DirBuff_DRV]
673 0000A627 80EB41              <1>     sub     bl, 'A'
674 0000A62A 38DF                <1>     cmp     bh, bl
675 0000A62C 750E                <1>     jne     short loc_lcdf_reload_current_dir1
676 0000A62E 8B15[6E7E0100]      <1>     mov     edx, [DirBuff_Cluster]
677 0000A634 3B15[3C770100]      <1>     cmp     edx, [Current_Dir_FCluster]
678 0000A63A 7412                <1>     je      short loc_cdir_locatefile_search
679                              <1>
680                              <1> loc_lcdf_reload_current_dir1:
681 0000A63C 30DB                <1>     xor     bl, bl
682                              <1> loc_lcdf_reload_current_dir2:
683 0000A63E 89DE                <1>     mov     esi, ebx
684 0000A640 81C600010900        <1>         add     esi, Logical_DOSDisks
685 0000A646 E866000000          <1>     call    reload_current_directory
686 0000A64B 734F                <1>     jnc     short loc_locatefile_search_again
687 0000A64D C3                  <1>     retn
688                              <1>
689                              <1> loc_cdir_locatefile_search:
690 0000A64E 31DB                <1>     xor     ebx, ebx
691 0000A650 55                  <1>     push    ebp ; 20/11/2017
692 0000A651 E87D000000          <1>     call    find_directory_entry
693 0000A656 5D                  <1>     pop     ebp ; 20/11/2017
694 0000A657 733B                <1>     jnc     short loc_cdir_locate_file_retn
695                              <1>
696                              <1> loc_locatefile_check_stc_reason:
697 0000A659 08ED                <1>     or      ch, ch
698 0000A65B 7436                <1>     jz      short loc_cdir_locate_file_stc_retn
699                              <1>
700                              <1> loc_locatefile_check_next_entryblock:
701                              <1>     ;mov    bh, [Current_Drv]
702                              <1>     ;sub    bl, bl
703                              <1>     ;movzx  esi, bx
704                              <1>     ; 28/07/2022
705 0000A65D 31DB                <1>         xor ebx, ebx
706 0000A65F 8A3D[42770100]      <1>     mov     bh, [Current_Drv]
707 0000A665 89DE                <1>     mov     esi, ebx
708 0000A667 81C600010900        <1>     add     esi, Logical_DOSDisks
709                              <1>
710 0000A66D 803D[40770100]00    <1>     cmp     byte [Current_Dir_Level], 0
711                              <1> ; 19/12/2025
712                              <1> %if 0
713                              <1>     jna     short loc_locatefile_check_FAT_type
714                              <1>
715                              <1>     cmp     byte [Current_FATType], 1
716                              <1>     jnb     short loc_locatefile_load_subdir_cluster
717                              <1>     retn
718                              <1> %else
719                              <1>     ; 19/12/2025
720 0000A674 7709                <1>     ja      short loc_locatefile_load_subdir_cluster
721                              <1> %endif
722                              <1>
723                              <1> loc_locatefile_check_FAT_type:
724 0000A676 803D[41770100]03    <1>     cmp     byte [Current_FATType], 3
725 0000A67D 7215                <1>     jb      short loc_cdir_locate_file_retn
```

```
                                <1>
                                <1> loc_locatefile_load_subdir_cluster:
0000A67F A1[6E7E0100]           <1>     mov     eax, [DirBuff_Cluster]
0000A684 E8F0180000             <1>     call    get_next_cluster
0000A689 730A                   <1>     jnc     short loc_locatefile_next_cluster
0000A68B 09C0                   <1>     or      eax, eax
0000A68D 7504                   <1>     jnz     short loc_locatefile_drive_not_ready_read_err
                                <1>     ;stc
                                <1>     ; 28/07/2022
                                <1> ;loc_locatefile_file_notfound:
                                <1>     ;mov    eax, 2 ; File/Directory/VolName not found
                                <1>     ;xor    eax, eax
                                <1>     ; 26/08/2024
                                <1>     ; eax = 0
0000A68F B002                   <1>     mov     al, 2
0000A691 F9                     <1>     stc
0000A692 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:
0000A693 F5                     <1>     cmc ;stc
                                <1> loc_cdir_locate_file_retn:
0000A694 C3                     <1>     retn
                                <1>
                                <1> loc_locatefile_next_cluster:
0000A695 E89B1A0000             <1>     call    load_FAT_sub_directory
                                <1>     ;jc     short loc_locatefile_drive_not_ready_read_err
0000A69A 72F8                   <1>     jc      short loc_cdir_locate_file_retn
                                <1>
                                <1> loc_locatefile_search_again:
0000A69C 8B35[287F0100]         <1>     mov     esi, [CDLF_FNAddress]
0000A6A2 66A1[267F0100]         <1>     mov     ax, [CDLF_AttributesMask]
0000A6A8 668B0D[2C7F0100]       <1>     mov     cx, [CDLF_DEType]
0000A6AF EB9D                   <1>     jmp     short loc_cdir_locatefile_search
                                <1>
                                <1> reload_current_directory:
                                <1>     ; 19/12/2025 (TRDOS 386 Kernel v2.0.10)
                                <1>     ; 28/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <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]
0000A6B1 A0[41770100]           <1>     mov     al, [Current_FATType]
0000A6B6 3C02                   <1>     cmp     al, 2
0000A6B8 770F                   <1>     ja      short loc_reload_FAT_sub_directory
0000A6BA 8A25[40770100]         <1>     mov     ah, [Current_Dir_Level]
                                <1> ; 19/12/2025
                                <1> %if 0
                                <1>     or      al, al
                                <1>     jz      short loc_reload_FS_directory
                                <1> %endif
0000A6C0 08E4                   <1>     or      ah, ah
0000A6C2 7505                   <1>     jnz     short loc_reload_FAT_sub_directory
                                <1> loc_reload_FAT_12_16_root_directory:
                                <1>     ;call   load_FAT_root_directory
                                <1>     ;retn
                                <1>     ; 28/07/2022
0000A6C4 E9EE190000             <1>     jmp     load_FAT_root_directory
                                <1>
                                <1> ; 19/12/2025
                                <1> %if 0
                                <1> loc_reload_FS_directory:
                                <1>     and     ah, ah
                                <1>     jnz     short loc_reload_FS_sub_directory
                                <1> loc_reload_FS_root_directory:
                                <1>     ;call   load_FS_root_directory
                                <1>     ;retn
                                <1>     ; 28/07/2022
                                <1>     jmp     load_FS_root_directory
                                <1> loc_reload_FS_sub_directory:
                                <1>     mov     eax, [Current_Dir_FCluster]
                                <1>     ;call   load_FS_sub_directory
                                <1>     ;retn
                                <1>     jmp     load_FS_sub_directory
                                <1> %endif
                                <1>
                                <1> loc_reload_FAT_sub_directory:
0000A6C9 A1[3C770100]           <1>     mov     eax, [Current_Dir_FCluster]
                                <1>     ;call   load_FAT_sub_directory
                                <1>     ;retn
                                <1>     ; 28/07/2022
0000A6CE E9621A0000             <1>     jmp     load_FAT_sub_directory
                                <1>
                                <1> find_directory_entry:
                                <1>     ; 28/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <1>     ; 02/03/2021 (TRDOS 386 v2.0.3) ((BugFix))
                                <1>     ; 14/02/2016
                                <1>     ; 13/02/2016
                                <1>     ; 10/02/2016
                                <1>     ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>     ; 14/08/2010 (DIR.ASM, "proc_find_direntry")
                                <1>     ; 19/09/2009
                                <1>     ; 2005
                                <1>     ; INPUT ->
                                <1>     ;       ESI = Sub Dir or File Name Address
                                <1>     ;       AL = Attributes Mask
                                <1>     ;       (<AL AND EntryAttrib> must be equal to AL)
                                <1>     ;       AH = Negative Attributes Mask (If AH>0)
                                <1>     ;       (<AH AND EntryAttrib> must be ZERO)
                                <1>     ;       CH > 0 Find First Free Dir Entry or Deleted Entry
                                <1>     ;       CL = 0 -> Return the First Free Dir Entry
                                <1>     ;       CL = E5h -> Return the 1st deleted entry
                                <1>     ;       CL = FFh -> Return the 1st deleted or free entry
                                <1>     ;       CL > 0 and CL <> E5h and CL <> FFh -> Return the first
                                <1>     ;              proper entry (which fits with Atributes Masks)
                                <1>     ;       CX = 0 -> Find Valid File/Directory/VolumeName
                                <1>     ;       ? = Any One Char
                                <1>     ;       * = Every Chars
                                <1>     ;       EBX = Current Dir Entry (BX)
                                <1>     ;
                                <1>     ; OUTPUT ->
                                <1>     ;       EDI = Directory Entry Address (in DirectoryBuffer)
                                <1>     ;       ESI = Sub Dir or File Name Address
                                <1>     ;       CF = 0 -> No Error, Proper Entry,
                                <1>     ;       DL = Attributes
                                <1>     ;       DH = Previous Entry Attr (LongName Check)
                                <1>     ;       AL > 0 -> Ambiguous filename wildcard "?" used
                                <1>     ;       AH > 0 -> Ambiguous filename wildcard "*" used
                                <1>     ;       AX = 0 -> Filename full fits with directory entry
```

```
                        <1>     ;       EBX = CurrentDirEntry (BX)
                        <1>     ;       CH = The 1st Name Char of Current Dir Entry
                        <1>     ;       CF = 1 -> Proper entry not found, Error Code in AX/AL
                        <1>     ;       CL = 0 and CH = 0 -> Free Entry (End Of Dir)
                        <1>     ;       CL = 0 and CH = E5h -> Deleted Entry fits with filters
                        <1>     ;       CL > 0 -> Entry not found, CH invalid
                        <1>     ;
                        <1>     ; (EAX, EBX, ECX, EDX, EDI, EBP will be changed)
                        <1>
0000A6D3 663B1D[6C7E0100] <1>     cmp     bx, [DirBuff_LastEntry]
0000A6DA 7728           <1>     ja      short loc_ffde_stc_retn_255 ; 28/07/2022
                        <1>
                        <1>     ;mov    [DirBuff_CurrentEntry], bx
                        <1>
0000A6DC BF00000800     <1>     mov     edi, Directory_Buffer
0000A6E1 66A3[387F0100] <1>     mov     [FDE_AttrMask], ax
                        <1>
0000A6E7 29C0           <1>     sub     eax, eax
                        <1>
                        <1>     ;;mov   [PreviousAttr], al ; 0 ;; 13/02/2016
0000A6E9 66A3[3A7F0100] <1>     mov     [AmbiguousFileName], ax ; 0
                        <1>
0000A6EF 6689D8         <1>     mov     ax, bx
                        <1>     ;shl    ax, 5 ; ; * 32 ; Directory entry size
                        <1>     ; 28/07/2022
0000A6F2 C1E005         <1>     shl     eax, 5
0000A6F5 01C7           <1>     add     edi, eax
                        <1>
0000A6F7 08ED           <1>     or      ch, ch
                        <1>     ;jnz    loc_find_free_deleted_entry_0
                        <1>     ; 28/07/2022
0000A6F9 7405           <1>     jz      short loc_fde_any_valid_entry_opt
0000A6FB E911010000     <1>     jmp     loc_find_free_deleted_entry_0
                        <1>
                        <1> loc_fde_any_valid_entry_opt:
0000A700 08C9           <1>     or      cl, cl
                        <1>         ;jnz        loc_ffde_stc_retn_255
                        <1>     ; 28/07/2022
0000A702 742E           <1>     jz      short check_find_dir_entry
                        <1>
                        <1>     ; 28/07/2022
                        <1> loc_ffde_stc_retn_255:
                        <1>     ; 02/03/2021 (TRDOS 386 v2.0.3) ((BugFix))
                        <1>     ; (ECX must not be > 65535)
                        <1>     ; ((because 'loc_ccd_save_current_dir'
                        <1>     ;  sets CX to 32 for 'rep movsd'))
0000A704 66B9FFFF       <1>     mov     cx, 0FFFFh
                        <1>     ;xor    ecx, ecx
                        <1>     ;dec    ecx ; 0FFFFFFFFh
                        <1>     ;xor    eax, eax
                        <1> loc_find_direntry_stc_retn:
                        <1> loc_check_ffde_retn_1:
                        <1>     ;mov    ax, 2
                        <1>     ;mov    eax, 2 ; File Not Found
                        <1>     ; 28/07/2022
0000A708 29C0           <1>     sub     eax, eax
0000A70A B002           <1>     mov     al, 2
0000A70C 8A35[3C7F0100] <1>     mov     dh, [PreviousAttr]
0000A712 66891D[6A7E0100] <1>     mov     [DirBuff_CurrentEntry], bx
0000A719 F9             <1>     stc
0000A71A C3             <1>     retn
                        <1>
                        <1>     ; 28/07/2022
                        <1> loc_find_dir_next_entry_prevdeleted:
0000A71B 80CA80         <1>     or      dl, 80h  ; Bit 7 -> deleted entry sign
                        <1>     ;jmp    short loc_find_dir_next_entry
                        <1>
                        <1>     ; 28/07/2022
                        <1> loc_find_dir_next_entry:
0000A71E 8815[3C7F0100] <1>     mov     byte [PreviousAttr], dl ; LongName check
                        <1> loc_find_dir_next_entry_1:
0000A724 5E             <1>     pop     esi
0000A725 83C720         <1>     add     edi, 32
                        <1>     ;;inc   word [DirBuff_EntryCounter]
                        <1>     ;inc    bx
                        <1>     ; 28/07/2022
0000A728 43             <1>     inc     ebx
0000A729 663B1D[6C7E0100] <1>     cmp     bx, [DirBuff_LastEntry]
0000A730 77D2           <1>     ja      short loc_ffde_stc_retn_255
                        <1>         ; 28/07/2022
                        <1>     ;jmp    short check_find_dir_entry
                        <1>
                        <1> check_find_dir_entry:
0000A732 66A1[387F0100] <1>     mov     ax, [FDE_AttrMask]
0000A738 8A2F           <1>     mov     ch, [edi]
0000A73A 80FD00         <1>     cmp     ch, 0 ; Is it never used entry?
                        <1>     ;jna    loc_find_direntry_stc_retn
                        <1>     ; 28/07/2022
0000A73D 7702           <1>     ja      short loc_fde_check_attrib
                        <1>     ; end of directory entries
0000A73F EBC7           <1>     jmp     loc_find_direntry_stc_retn
                        <1> loc_fde_check_attrib:
0000A741 56             <1>     push    esi
0000A742 8A570B         <1>     mov     dl, [edi+0Bh] ; File attributes
0000A745 80FDE5         <1>     cmp     ch, 0E5h ; Is it a deleted file?
0000A748 74D1           <1>     je      short loc_find_dir_next_entry_prevdeleted
                        <1>
0000A74A 80FA0F         <1>     cmp     dl, 0Fh ; longname sub component check
0000A74D 7505           <1>     jne     short loc_check_attributes_mask
0000A74F E8A6010000     <1>     call    save_longname_sub_component
                        <1>
                        <1> loc_check_attributes_mask:
0000A754 88C6           <1>     mov     dh, al
0000A756 20D6           <1>     and     dh, dl
                        <1>     ; 28/07/2022
0000A758 38F0           <1>     cmp     al, dh
0000A75A 75C2           <1>     jne     short loc_find_dir_next_entry
0000A75C 20D4           <1>     and     ah, dl
0000A75E 75BE           <1>         jnz short loc_find_dir_next_entry
0000A760 80FA0F         <1>     cmp     dl, 0Fh
0000A763 7515           <1>     jne     short pass_direntry_attr_check
                        <1>
0000A765 3C0F           <1>     cmp     al, 0Fh ; AL = 0Fh -> find long name
0000A767 75B5           <1>     jne     short loc_find_dir_next_entry
                        <1>
0000A769 5E             <1>     pop     esi
                        <1>     ;xor    ax, ax
                        <1>     ; 28/07/2022
                        <1>     ;sub    eax, eax
0000A76A 30C0           <1>     xor     al, al
0000A76C 8A35[3C7F0100] <1>     mov     dh, [PreviousAttr]
0000A772 66891D[6A7E0100] <1>     mov     [DirBuff_CurrentEntry], bx
0000A779 C3             <1>     retn
                        <1>
```

```
                                <1> pass_direntry_attr_check:
0000A77A 89FD                   <1>     mov     ebp, edi ; 14/02/2016
                                <1>     ;mov    ecx, 8
                                <1>     ; 28/07/2022
0000A77C 29C9                   <1>     sub     ecx, ecx
0000A77E B108                   <1>     mov     cl, 8
                                <1> loc_lodsb_find_dir:
0000A780 AC                     <1>     lodsb
0000A781 3C2A                   <1>     cmp     al, '*'
0000A783 7508                   <1>     jne     short pass_fde_ambiguous1_check
0000A785 FE05[3B7F0100]         <1>         inc     byte [AmbiguousFileName+1]
0000A78B EB23                   <1>     jmp     short loc_check_direntry_extension
                                <1>
                                <1> pass_fde_ambiguous1_check:
0000A78D 3C3F                   <1>     cmp     al, '?'
0000A78F 750D                   <1>     jne     short pass_fde_ambiguous2_check
0000A791 FE05[3A7F0100]         <1>     inc     byte [AmbiguousFileName]
0000A797 803F20                 <1>     cmp     byte [edi], 20h
0000A79A 763E                   <1>     jna     short loc_find_dir_next_entry_ebp
0000A79C EB0F                   <1>     jmp     short loc_scasb_find_dir_inc_di
                                <1>
                                <1> pass_fde_ambiguous2_check:
0000A79E 3C20                   <1>     cmp     al, 20h
0000A7A0 7507                   <1>     jne     short loc_scasb_find_dir
0000A7A2 803F20                 <1>     cmp     byte [edi], 20h
0000A7A5 7533                   <1>     jne     short loc_find_dir_next_entry_ebp
0000A7A7 EB07                   <1>     jmp     short loc_check_direntry_extension
                                <1>
                                <1> loc_scasb_find_dir:
0000A7A9 3A07                   <1>     cmp     al, [edi]
0000A7AB 752D                   <1>     jne     short loc_find_dir_next_entry_ebp
                                <1> loc_scasb_find_dir_inc_di:
0000A7AD 47                     <1>     inc     edi
0000A7AE E2D0                   <1>     loop    loc_lodsb_find_dir
                                <1>
                                <1> loc_check_direntry_extension:
0000A7B0 BE08000000             <1>     mov     esi, 8
0000A7B5 89F7                   <1>     mov     edi, esi ; 8
0000A7B7 033424                 <1>     add     esi, [esp] ; Sub Dir or File Name Address
0000A7BA 01EF                   <1>     add     edi, ebp
0000A7BC B103                   <1>     mov     cl, 3
                                <1> loc_lodsb_find_dir_ext:
0000A7BE AC                     <1>     lodsb
0000A7BF 3C2A                   <1>     cmp     al, '*'
0000A7C1 7508                   <1>     jne     short pass_fde_ambiguous3_check
0000A7C3 FE05[3B7F0100]         <1>     inc     byte [AmbiguousFileName+1]
0000A7C9 EB1F                   <1>     jmp     short loc_find_dir_proper_direntry
                                <1>
                                <1> pass_fde_ambiguous3_check:
0000A7CB 3C3F                   <1>     cmp     al, '?'
0000A7CD 7512                   <1>     jne     short pass_fde_ambiguous4_check
0000A7CF FE05[3A7F0100]         <1>     inc     byte [AmbiguousFileName]
0000A7D5 803F20                 <1>     cmp     byte [edi], 20h
                                <1>     ;jna    short loc_find_dir_next_entry_ebp
                                <1>     ;jmp    short loc_scasb_find_dir_ext_inc_di
                                <1>     ; 28/07/2022
0000A7D8 7732                   <1>     ja      short loc_scasb_find_dir_ext_inc_di
                                <1>
                                <1> loc_find_dir_next_entry_ebp:
0000A7DA 89EF                   <1>     mov     edi, ebp ; 14/02/2016
0000A7DC E93DFFFFFF             <1>     jmp     loc_find_dir_next_entry ; 28/07/2022
                                <1>
                                <1> pass_fde_ambiguous4_check:
0000A7E1 3C20                   <1>     cmp     al, 20h
0000A7E3 7523                   <1>     jne     short loc_scasb_find_dir_ext
0000A7E5 803F20                 <1>     cmp     byte [edi], 20h
                                <1>     ; 28/07/2022
0000A7E8 75F0                   <1>     jne     short loc_find_dir_next_entry_ebp
                                <1>     ;jmp    short loc_find_dir_proper_direntry
                                <1>
                                <1> loc_find_dir_proper_direntry:
0000A7EA 30C9                   <1>     xor     cl, cl
                                <1> loc_find_dir_proper_direntry_1:
0000A7EC 5E                     <1>     pop     esi
0000A7ED 89EF                   <1>         mov     edi, ebp
0000A7EF 8A2F                   <1>     mov     ch, [edi]
0000A7F1 8A570B                 <1>     mov     dl, [edi+0Bh] ; Dir entry attributes
0000A7F4 66A1[3A7F0100]         <1>     mov     ax, [AmbiguousFileName]
                                <1> loc_find_dir_proper_direntry_2:
0000A7FA 8A35[3C7F0100]         <1>     mov     dh, [PreviousAttr]
0000A800 66891D[6A7E0100]       <1>     mov     [DirBuff_CurrentEntry], bx
0000A807 C3                     <1>     retn
                                <1>
                                <1> loc_scasb_find_dir_ext:
0000A808 3A07                   <1>     cmp     al, [edi]
0000A80A 75CE                   <1>     jne     short loc_find_dir_next_entry_ebp
                                <1> loc_scasb_find_dir_ext_inc_di:
0000A80C 47                     <1>     inc     edi
0000A80D E2AF                   <1>     loop    loc_lodsb_find_dir_ext
0000A80F EBDB                   <1>     jmp     short loc_find_dir_proper_direntry_1
                                <1>
                                <1> loc_find_free_deleted_entry_0:
0000A811 66A1[387F0100]         <1>     mov     ax, [FDE_AttrMask]
0000A817 8A2F                   <1>     mov     ch, [edi]
0000A819 8A570B                 <1>     mov     dl, [edi+0Bh] ; File attributes
0000A81C 08C9                   <1>     or      cl, cl
0000A81E 7407                   <1>     jz      short loc_check_ffde_0_repeat
                                <1>     ;cmp    cl, 0E5h
                                <1>     ;je     short pass_loc_check_ffde_0_err
0000A820 80F9FF                 <1>     cmp     cl, 0FFh
0000A823 7430                   <1>     je      short loc_find_free_deleted_entry_1
0000A825 EB4A                   <1>     jmp     short pass_loc_check_ffde_0_err
                                <1>
                                <1> loc_check_ffde_0_repeat:
0000A827 08ED                   <1>     or      ch, ch
0000A829 7510                   <1>     jnz     short loc_check_ffde_0_next
                                <1>
                                <1> loc_check_ffde_retn_2:
                                <1>     ;sub    ax, ax
                                <1>     ; 28/07/2022
0000A82B 29C0                   <1>     sub     eax, eax
0000A82D 8A35[3C7F0100]         <1>     mov     dh, [PreviousAttr]
0000A833 66891D[6A7E0100]       <1>     mov     [DirBuff_CurrentEntry], bx
0000A83A C3                     <1>     retn
                                <1>
                                <1> loc_check_ffde_0_next:
                                <1>     ;inc    bx
                                <1>     ; 28/07/2022
0000A83B 43                     <1>     inc     ebx
0000A83C 83C720                 <1>     add     edi, 32
                                <1>     ;inc    word [DirBuff_EntryCounter]
                                <1>
0000A83F 663B1D[6C7E0100]       <1>         cmp bx, [DirBuff_LastEntry]
                                <1>     ;ja     short loc_ffde_stc_retn_255
```

```
1098                                <1>     ; 07/08/2022
1099 0000A846 773A                  <1>     ja      short jmp_ffde_stc_retn_255
1100                                <1>
1101 0000A848 8815[3C7F0100]        <1>     mov     [PreviousAttr], dl
1102 0000A84E 8A2F                  <1>     mov     ch, [edi]
1103 0000A850 8A570B                <1>     mov     dl, [edi+0Bh] ; file attributes
1104 0000A853 EBD2                  <1>     jmp     short loc_check_ffde_0_repeat
1105                                <1>
1106                                <1> loc_find_free_deleted_entry_1:
1107 0000A855 28D2                  <1>     sub     dl, dl
1108                                <1> loc_find_free_deleted_entry_2:
1109 0000A857 20ED                  <1>     and     ch, ch
1110 0000A859 74D0                  <1>     jz      short loc_check_ffde_retn_2
1111 0000A85B 80FDE5                <1>     cmp     ch, 0E5h
1112 0000A85E 74CB                  <1>     je      short loc_check_ffde_retn_2
1113                                <1>     ;inc    bx
1114                                <1>     ; 28/07/2022
1115 0000A860 43                    <1>     inc     ebx
1116 0000A861 83C720                <1>     add     edi, 32
1117 0000A864 663B1D[6C7E0100]      <1>     cmp     bx, [DirBuff_LastEntry]
1118                                <1>     ;ja     short loc_ffde_stc_retn_255
1119                                <1>     ; 07/08/2022
1120 0000A86B 7715                  <1>     ja      short jmp_ffde_stc_retn_255
1121                                <1>
1122 0000A86D 8A2F                  <1>     mov     ch, [edi]
1123 0000A86F EBE6                  <1>     jmp     short loc_find_free_deleted_entry_2
1124                                <1>
1125                                <1> pass_loc_check_ffde_0_err:
1126 0000A871 38CD                  <1>     cmp     ch, cl
1127 0000A873 741F                  <1>     je      short loc_check_ffde_attrib
1128                                <1>
1129                                <1>     ;inc    bx
1130                                <1>     ; 28/07/2022
1131 0000A875 43                    <1>     inc     ebx
1132 0000A876 83C720                <1>     add     edi, 32
1133 0000A879 663B1D[6C7E0100]      <1>     cmp     bx, [DirBuff_LastEntry]
1134                                <1>         ;ja loc_ffde_stc_retn_255
1135                                <1>     ; 28/07/2022
1136 0000A880 7605                  <1>     jna     short loc_ffe_save_prev_attr
1137                                <1> jmp_ffde_stc_retn_255:     ; 07/08/2022
1138 0000A882 E97DFEFFFF            <1>     jmp     loc_ffde_stc_retn_255
1139                                <1>
1140                                <1> loc_ffe_save_prev_attr: ; 28/07/2022
1141 0000A887 8815[3C7F0100]        <1>     mov     [PreviousAttr], dl
1142 0000A88D 8A2F                  <1>     mov     ch, [edi]
1143 0000A88F 8A570B                <1>     mov     dl, [edi+0Bh]
1144 0000A892 EBDD                  <1>     jmp     short pass_loc_check_ffde_0_err
1145                                <1>
1146                                <1> loc_check_ffde_attrib:
1147 0000A894 88C6                  <1>     mov     dh, al
1148 0000A896 20D6                  <1>     and     dh, dl
1149 0000A898 38F0                  <1>     cmp     al, dh
1150 0000A89A 759F                  <1>     jne     short loc_check_ffde_0_next
1151 0000A89C 20D4                  <1>     and     ah, dl
1152 0000A89E 759B                  <1>     jnz     short loc_check_ffde_0_next
1153 0000A8A0 30C9                  <1>     xor     cl, cl
1154 0000A8A2 EB87                  <1>         jmp short loc_check_ffde_retn_2
1155                                <1>
1156                                <1> convert_file_name:
1157                                <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
1158                                <1>     ; 06/03/2016
1159                                <1>     ; 11/02/2016
1160                                <1>     ; 07/02/2016 (TRDOS 386 = TRDOS v2.0)
1161                                <1>     ; 06/10/2009
1162                                <1>     ; 2005
1163                                <1>     ;
1164                                <1>     ; INPUT  ->
1165                                <1>     ;       ESI = Dot File Name Location
1166                                <1>     ;       EDI = Dir Entry Format File Name Location
1167                                <1>     ; OUTPUT ->
1168                                <1>     ;       EDI = Dir Entry Format File Name Location
1169                                <1>     ;       ESI = Dot File Name Location (capitalized)
1170                                <1>     ;
1171                                <1>     ; (ECX, AL will be changed)
1172                                <1>
1173 0000A8A4 56                    <1>     push    esi
1174 0000A8A5 57                    <1>     push    edi
1175                                <1>
1176                                <1>     ;mov    ecx, 11
1177                                <1>     ; 29/07 2022
1178 0000A8A6 29C9                  <1>     sub     ecx, ecx
1179 0000A8A8 B10B                  <1>     mov     cl, 11
1180 0000A8AA B020                  <1>     mov     al, 20h
1181 0000A8AC F3AA                  <1>     rep     stosb
1182                                <1>
1183 0000A8AE 8B3C24                <1>     mov     edi, [esp]
1184                                <1>
1185 0000A8B1 B10C                  <1>     mov     cl, 12 ; file name length (max.)
1186                                <1>     ; 06/03/2016
1187                                <1>     ; Directory entry name limit (11 bytes) check for
1188                                <1>     ; 'rename_directory_entry' procedure.
1189                                <1>     ; (EDI points to Directory Entry)
1190                                <1>     ; (If the file name would not contain a dot
1191                                <1>     ; and file name length would be 12, this would cause to
1192                                <1>     ; overwrite the attributes byte of the directory entry.)
1193                                <1>     ;
1194 0000A8B3 B50B                  <1>     mov     ch, 11 ; directory entry's name length
1195                                <1> loc_check_first_dot:
1196 0000A8B5 8A06                  <1>     mov     al, [esi]
1197 0000A8B7 3C2E                  <1>     cmp     al, 2Eh
1198 0000A8B9 750C                  <1>     jne     short pass_check_first_dot
1199 0000A8BB 8807                  <1>     mov     [edi], al
1200 0000A8BD 47                    <1>     inc     edi
1201 0000A8BE 46                    <1>     inc     esi
1202 0000A8BF FEC9                  <1>     dec     cl
1203 0000A8C1 75F2                  <1>     jnz     short loc_check_first_dot
1204                                <1>     ;;(ecx <= 12)
1205                                <1>     ;;loop  loc_check_first_dot
1206 0000A8C3 EB30                  <1>     jmp     short stop_convert_file
1207                                <1>
1208                                <1> loc_get_fchar:
1209 0000A8C5 8A06                  <1>     mov     al, [esi]
1210                                <1> pass_check_first_dot:
1211 0000A8C7 3C61                  <1>     cmp     al, 61h ; 'a'
1212 0000A8C9 7208                  <1>     jb      short pass_name_capitalize
1213 0000A8CB 3C7A                  <1>     cmp     al, 7Ah ; 'z'
1214 0000A8CD 7704                  <1>     ja      short pass_name_capitalize
1215 0000A8CF 24DF                  <1>     and     al, 0DFh
1216 0000A8D1 8806                  <1>     mov     [esi], al
1217                                <1> pass_name_capitalize:
1218 0000A8D3 3C21                  <1>     cmp     al, 21h
1219 0000A8D5 721E                  <1>     jb      short stop_convert_file
1220 0000A8D7 3C2E                  <1>     cmp     al, 2Eh ; '.'
1221 0000A8D9 750C                  <1>     jne     short pass_dot_space
```

```
1222                                <1> add_dot_space:
1223 0000A8DB 80F904               <1>     cmp     cl, 4
1224 0000A8DE 760E                 <1>     jna     short inc_and_loop
1225 0000A8E0 47                   <1>     inc     edi
1226 0000A8E1 FECD                 <1>     dec     ch ; 06/03/2016
1227 0000A8E3 FEC9                 <1>     dec     cl
1228 0000A8E5 EBF4                 <1>     jmp     short add_dot_space
1229                                <1>
1230                                <1>     ;mov    al, 4
1231                                <1>     ;cmp    cl, al
1232                                <1>     ;jna    short inc_and_loop
1233                                <1>     ;sub    cl, al
1234                                <1>     ;add    edi, ecx
1235                                <1>     ;mov    cl, al
1236                                <1>     ;jmp    short inc_and_loop
1237                                <1>
1238                                <1> pass_dot_space:
1239 0000A8E7 8807                 <1>     mov     [edi], al
1240                                <1> loc_after_double_dot:
1241                                <1>     ; 06/03/2016
1242 0000A8E9 FECD                 <1>     dec     ch ; count down for 11 bytes dir entry limit
1243 0000A8EB 740A                 <1>     jz      short stop_convert_file_x
1244 0000A8ED 47                   <1>     inc     edi
1245                                <1> inc_and_loop:
1246 0000A8EE FEC9                 <1>     dec     cl ; count down for 12 bytes filename limit
1247 0000A8F0 7403                 <1>     jz      short stop_convert_file
1248 0000A8F2 46                   <1>     inc     esi
1249                                <1>     ;;(ecx <= 12)
1250                                <1>     ;;loop  loc_get_fchar
1251 0000A8F3 EBD0                 <1>     jmp     short loc_get_fchar
1252                                <1>
1253                                <1> stop_convert_file:
1254                                <1>     ; 06/03/2016
1255 0000A8F5 30ED                 <1>     xor     ch, ch
1256                                <1>     ; ECX < 256 ; 'find_first_file' -> xor cl, cl
1257                                <1> stop_convert_file_x:
1258 0000A8F7 5F                   <1>     pop     edi
1259 0000A8F8 5E                   <1>     pop     esi
1260 0000A8F9 C3                   <1>     retn
1261                                <1>
1262                                <1> save_longname_sub_component:
1263                                <1>     ; 13/02/2016
1264                                <1>     ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
1265                                <1>     ; 28/02/2010
1266                                <1>     ; 17/10/2009
1267                                <1>     ; INPUT ->
1268                                <1>     ;       EDI = Directory Entry
1269                                <1>     ;       // This procedure is called
1270                                <1>     ;       // from 'find_directory_entry' procedure.
1271                                <1>     ;       // If the last entry returns with
1272                                <1>     ;       // a non-zero LongnameFound value and
1273                                <1>     ;       // if LFN_CheckSum value is equal to
1274                                <1>     ;       // the next shortname checksum,
1275                                <1>     ;       // long name is valid.
1276                                <1>     ;       // If a longname is longer than 65 bytes,
1277                                <1>     ;       // it is invalid for trdos. (>45h)
1278                                <1>
1279 0000A8FA 57                   <1>     push    edi
1280 0000A8FB 56                   <1>     push    esi
1281                                <1>     ;push   ebx
1282                                <1>     ;push   ecx
1283                                <1>     ;push   edx
1284 0000A8FC 50                   <1>     push    eax
1285                                <1>
1286 0000A8FD 29C9                 <1>     sub     ecx, ecx
1287                                <1>     ;sub    eax, eax
1288 0000A8FF B11A                 <1>     mov     cl, 26
1289                                <1>
1290 0000A901 0FB607               <1>     movzx   eax, byte [edi] ; LDIR_Order
1291 0000A904 3C41                 <1>     cmp     al, 41h  ; 40h (last long entry sign) + 1
1292 0000A906 722B                 <1>     jb      short pass_pslnsc_last_long_entry
1293                                <1>
1294 0000A908 88C4                 <1>     mov     ah, al
1295 0000A90A 80EC40               <1>     sub     ah, 40h
1296 0000A90D 8825[3E7F0100]       <1>     mov     [LFN_EntryLength], ah
1297                                <1>
1298 0000A913 3C45                 <1>     cmp     al, 45h  ; 40h (last long entry sign) + 5
1299                                <1>             ; Max 130 byte length is usable in TRDOS
1300                                <1> ; 26*5 = 130
1301 0000A915 7753                 <1>     ja      short loc_pslnsc_retn
1302                                <1>
1303 0000A917 2407                 <1>     and     al, 07h ; 0Fh
1304 0000A919 A2[3D7F0100]         <1>     mov     [LongNameFound], al
1305                                <1>
1306 0000A91E FEC8                 <1>     dec     al
1307                                <1>     ;mov    cl, 26
1308 0000A920 F6E1                 <1>     mul     cl
1309                                <1>
1310 0000A922 89C6                 <1>     mov     esi, eax
1311 0000A924 01CE                 <1>     add     esi, ecx
1312                                <1>             ; to make is an ASCIIZ string
1313                                <1>             ; with ax+26 bytes length
1314 0000A926 81C6[407F0100]       <1>     add     esi, LongFileName
1315 0000A92C 66C7060000           <1>     mov     word [esi], 0
1316 0000A931 EB16                 <1>     jmp     short loc_pslsc_move_ldir_name2
1317                                <1>
1318                                <1> pass_pslnsc_last_long_entry:
1319 0000A933 3C04                 <1>     cmp     al, 04h
1320 0000A935 7733                 <1>     ja      short loc_pslnsc_retn
1321 0000A937 FE0D[3D7F0100]       <1>     dec     byte [LongNameFound]
1322 0000A93D 3A05[3D7F0100]       <1>     cmp     al, [LongNameFound]
1323 0000A943 7525                 <1>     jne     short loc_pslnsc_retn
1324                                <1>
1325                                <1> loc_pslsc_move_ldir_name1:
1326 0000A945 FEC8                 <1>     dec     al
1327                                <1>     ;mov    cl, 26
1328 0000A947 F6E1                 <1>     mul     cl
1329                                <1>
1330                                <1> loc_pslsc_move_ldir_name2:
1331 0000A949 8A4F0D               <1>     mov     cl, [edi+0Dh] ; long name checksum
1332 0000A94C 880D[3F7F0100]       <1>     mov     [LFN_CheckSum], cl
1333 0000A952 89FE                 <1>     mov     esi, edi ; LDIR_Order
1334 0000A954 BF[407F0100]         <1>     mov     edi, LongFileName
1335 0000A959 01C7                 <1>     add     edi, eax
1336 0000A95B 46                   <1>     inc     esi
1337 0000A95C B105                 <1>     mov     cl, 5 ; chars 1 to 5
1338 0000A95E F366A5               <1>     rep     movsw
1339 0000A961 83C603               <1>     add     esi, 3
1340 0000A964 A5                   <1>     movsd  ; char 6 & 7
1341 0000A965 A5                   <1>     movsd  ; char 8 & 9
1342 0000A966 A5                   <1>     movsd  ; char 10 & 11
1343 0000A967 46                   <1>     inc     esi
1344 0000A968 46                   <1>     inc     esi
1345 0000A969 A5                   <1>     movsd   ; char 12 & 13
```

```
                                <1>
                                <1> loc_pslnsc_retn:
0000A96A 58                     <1>     pop     eax
                                <1>     ;pop    edx
                                <1>     ;pop    ecx
                                <1>     ;pop    ebx
0000A96B 5E                     <1>     pop     esi
0000A96C 5F                     <1>     pop     edi
                                <1>
0000A96D C3                     <1>     retn
                                <1>
                                <1> parse_path_name:
                                <1>     ; 02/12/2025 (TRDOS 386 v2.0.10, BugFix)
                                <1>     ; 03/09/2024 (TRDOS 386 v2.0.9)
                                <1>     ; 09/08/2022
                                <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <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
0000A96E 57                     <1>     push    edi
                                <1>     ;mov    ecx, 20  ; 80 bytes
                                <1>     ; 29/07/2022
0000A96F 29C9                   <1>     sub     ecx, ecx
0000A971 B114                   <1>     mov     cl, 20
0000A973 31C0                   <1>     xor     eax, eax
0000A975 F3AB                   <1>     rep     stosd
0000A977 5F                     <1>     pop     edi
                                <1>
0000A978 668B06                 <1>     mov     ax, [esi]
0000A97B 80FC3A                 <1>     cmp     ah, ':'
0000A97E 741C                   <1>     je      short loc_ppn_change_drive
0000A980 A0[42770100]           <1>     mov     al, [Current_Drv]
0000A985 EB33                   <1>     jmp     short pass_ppn_change_drive
                                <1>
                                <1> pass_ppn_cdir:
0000A987 8B35[62800100]         <1>     mov     esi, [First_Path_Pos]
0000A98D AC                     <1>     lodsb
                                <1> loc_ppn_get_filename:
0000A98E 83C741                 <1>     add     edi, 65 ; FindFile_Name location
                                <1>     ; TRDOS Filename length must not be more than 12 bytes
                                <1>     ;mov    ecx, 12
0000A991 B10C                   <1>     mov     cl, 12
                                <1> loc_ppn_get_fnchar_next:
0000A993 AA                     <1>     stosb
0000A994 AC                     <1>     lodsb
0000A995 3C21                   <1>     cmp     al, 21h
0000A997 7269                   <1>     jb      short loc_ppn_clc_return
0000A999 E2F8                   <1>         loop    loc_ppn_get_fnchar_next
                                <1> loc_ppn_return:
0000A99B C3                     <1>     retn
                                <1>
                                <1> loc_ppn_change_drive:
                                <1>     ; 29/07/2022
                                <1>     ; ecx = 0
0000A99C 24DF                   <1>     and     al, 0DFh
0000A99E 2C41                   <1>     sub     al, 'A' ; A:
0000A9A0 7264                   <1>     jc      short loc_ppn_invalid_drive
0000A9A2 3805[5C2E0100]         <1>     cmp     [Last_DOS_DiskNo], al
0000A9A8 725C                   <1>     jb      short loc_ppn_invalid_drive
                                <1>
0000A9AA 46                     <1>     inc     esi
0000A9AB 46                     <1>     inc     esi
0000A9AC 8A26                   <1>     mov     ah, [esi]
0000A9AE 80FC21                 <1>     cmp     ah, 21h
0000A9B1 7307                   <1>     jnb     short pass_ppn_change_drive
                                <1>
                                <1> loc_ppn_cmd_failed:
                                <1>     ; File or directory name is not existing
0000A9B3 8807                   <1>     mov     [edi], al ; Drv
0000A9B5 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!")
0000A9B9 C3                     <1>     retn
                                <1>
                                <1> pass_ppn_change_drive:
0000A9BA 8935[62800100]         <1>     mov     [First_Path_Pos], esi
                                <1>     ;mov    dword [Last_Slash_Pos], 0
                                <1>     ; 29/07/2022
0000A9C0 890D[66800100]         <1>     mov     [Last_Slash_Pos], ecx ; 0
0000A9C6 AA                     <1>     stosb
0000A9C7 8A06                   <1>     mov     al, [esi]
                                <1> loc_scan_ppn_dslash:
0000A9C9 3C2F                   <1>     cmp     al, '/'
0000A9CB 7506                   <1>     jne     short loc_scan_next_slash_pos
0000A9CD 8935[66800100]         <1>     mov     [Last_Slash_Pos], esi
                                <1> loc_scan_next_slash_pos:
0000A9D3 46                     <1>     inc     esi
0000A9D4 8A06                   <1>     mov     al, [esi]
0000A9D6 3C20                   <1>     cmp     al, 20h
0000A9D8 77EF                   <1>     ja      short loc_scan_ppn_dslash
                                <1>     ;cmp    dword [Last_Slash_Pos], 0
                                <1>     ; 09/08/2022
                                <1>     ;cmp    [Last_Slash_Pos], ecx ; 0 ?
                                <1>     ;jna    short pass_ppn_cdir
                                <1>
0000A9DA 8B0D[66800100]         <1>     mov     ecx, [Last_Slash_Pos]
                                <1>     ; 03/09/2024
                                <1>     ;jcxz   pass_ppn_cdir
                                <1>     ; 02/12/2025
0000A9E0 E3A5                   <1>     jecxz   pass_ppn_cdir
                                <1>
0000A9E2 8B35[62800100]         <1>     mov     esi, [First_Path_Pos]
0000A9E8 29F1                   <1>     sub     ecx, esi
0000A9EA 41                     <1>     inc     ecx
                                <1>     ;cmp    ecx, 64
0000A9EB 80F940                 <1>     cmp     cl, 64
0000A9EE 7715                   <1>     ja      short loc_ppn_invalid_drive_stc
                                <1>
```

```
1470 0000A9F0 89F8                <1>     mov     eax, edi ; Dest Dir String Location (65 bytes)
1471 0000A9F2 F3A4                <1>     rep     movsb
1472                              <1>     ;mov    [edi], cl ; 0, End of Dir String
1473 0000A9F4 8B35[66800100]      <1>     mov     esi, [Last_Slash_Pos]
1474 0000A9FA 46                  <1>     inc     esi
1475 0000A9FB 89C7                <1>     mov     edi, eax
1476 0000A9FD AC                  <1>     lodsb
1477 0000A9FE 3C21                <1>     cmp     al, 21h
1478 0000AA00 738C                <1>     jnb     short loc_ppn_get_filename
1479                              <1> loc_ppn_clc_return:
1480                              <1>     ;clc
1481 0000AA02 31C0                <1>     xor     eax, eax
1482 0000AA04 C3                  <1>     retn
1483                              <1>
1484                              <1> loc_ppn_invalid_drive_stc:
1485 0000AA05 F5                  <1>     cmc     ; stc
1486                              <1> loc_ppn_invalid_drive:
1487                              <1>     ; cf = 1
1488                              <1>     ; The Drive Letter/Char < "A" or > "Z"
1489 0000AA06 66B80F00            <1>     mov     ax, 0Fh
1490                              <1>     ; MS-DOS Error Code 0Fh = Disk Drive Invalid
1491                              <1>     ; (MainProg ErrMsg: "Drive not ready or read error!")
1492 0000AA0A C3                  <1>     retn
1493                              <1>
1494                              <1> find_longname:
1495                              <1>     ; 19/12/2025 (TRDOS 386 v2.0.10)
1496                              <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
1497                              <1>     ; 13/02/2016 (TRDOS 386 = TRDOS v2.0)
1498                              <1>     ; 24/01/2010 (DIR.ASM, 'proc_find_longname')
1499                              <1>     ; 17/10/2009
1500                              <1>
1501                              <1>     ; INPUT ->
1502                              <1>     ;       ESI = DOS short file name address
1503                              <1>     ;       for example: "filename.ext"
1504                              <1>     ;
1505                              <1>     ; OUTPUT ->
1506                              <1>     ;       ESI = ASCIIZ longname address (cf = 0)
1507                              <1>     ;       cf = 1 -> error number returns in EAX (AL)
1508                              <1>     ;       AL = 0 & CF=1 -> longname not found
1509                              <1>     ;            the file/directory has no longname
1510                              <1>     ;       cf = 0 -> AL = FAT Type
1511                              <1>
1512                              <1>     ; 17/10/2009
1513                              <1>     ; ASCIIZ string will be returned
1514                              <1>     ; as LongFileName
1515                              <1>     ; clearing/reset is not needed
1516                              <1>     ;mov    ecx, 33
1517                              <1>     ;mov    edi, LongFileName
1518                              <1>     ;sub    ax, ax ; 0
1519                              <1>     ;rep    stosw
1520                              <1>
1521                              <1>     ;mov    byte [LongNameFound], 0
1522                              <1>
1523                              <1>     ; ESI = ASCIIZ file/directory name address
1524                              <1>     ;   AL = Attributes AND mask
1525                              <1>     ;       (Result of AND must be equal to AL)
1526                              <1>     ;   AH = Negative attributes mask
1527                              <1>     ;       (Result of AND must be ZERO)
1528 0000AA0B 66B80008            <1>     mov     ax, 0800h
1529                              <1>             ; it must not be volume name or longname
1530 0000AA0F E856DFFFFF          <1>     call    find_first_file
1531 0000AA14 7207                <1>     jc      short loc_fln_retn
1532                              <1>
1533                              <1> ; 19/12/2025
1534                              <1> %if 0
1535                              <1> loc_fln_check_FAT_Type:
1536                              <1>     cmp     byte [Current_FATType], 1
1537                              <1>     jnb     short loc_fln_check_longname_yes_sign
1538                              <1>
1539                              <1>     ;call   get_fs_longname
1540                              <1>     ;retn
1541                              <1>     ; 29/07/2022
1542                              <1>     jmp     get_fs_longname
1543                              <1> %endif
1544                              <1>
1545                              <1> loc_fln_check_longname_yes_sign:
1546 0000AA16 08FF                <1>     or      bh, bh
1547 0000AA18 7504                <1>     jnz     short loc_fln_check_longnamefound_number
1548                              <1> loc_fln_longname_not_found_retn:
1549 0000AA1A 31C0                <1>     xor     eax, eax
1550                              <1>     ; cf = 1 & al = 0 -> longname not found
1551 0000AA1C F9                  <1>     stc
1552                              <1> loc_fln_retn:
1553 0000AA1D C3                  <1>     retn
1554                              <1>
1555                              <1> loc_fln_check_longnamefound_number:
1556                              <1>     ; 'LongNameFound' is set by
1557                              <1>         ; by 'save_longname_sub_component'
1558                              <1>     ; which is called from
1559                              <1>     ; 'find_directory_entry'
1560                              <1>     ; which is called from
1561                              <1>     ; 'find_first_file'
1562                              <1>     ; It must 1 if the longname is valid
1563 0000AA1E 803D[3D7F0100]01    <1>         cmp     byte [LongNameFound], 1
1564 0000AA25 75F3                <1>     jne     short loc_fln_longname_not_found_retn
1565                              <1>
1566                              <1> loc_fln_calculate_checksum:
1567 0000AA27 E813000000          <1>     call    calculate_checksum
1568                              <1>     ; AL = shortname checksum
1569                              <1>
1570                              <1> loc_fln_longname_validation:
1571                              <1>     ; 'LFN_CheckSum' has been set already
1572                              <1>     ; by 'save_longname_sub_component'
1573                              <1>     ; which is called from
1574                              <1>     ; 'find_directory_entry'
1575                              <1>     ; which is called from
1576                              <1>     ; 'find_first_file'
1577 0000AA2C 3805[3F7F0100]      <1>     cmp     [LFN_CheckSum], al
1578 0000AA32 75E6                <1>     jne     short loc_fln_longname_not_found_retn
1579                              <1>
1580 0000AA34 BE[407F0100]        <1>     mov     esi, LongFileName
1581 0000AA39 A0[41770100]        <1>     mov     al, [Current_FATType]
1582 0000AA3E C3                  <1>     retn
1583                              <1>
1584                              <1> calculate_checksum:
1585                              <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
1586                              <1>     ; 13/02/2016 (TRDOS 386 = TRDOS v2.0)
1587                              <1>     ; 17/10/2009 (DIR.ASM, 'proc_calculate_checksum')
1588                              <1>         ;
1589                              <1>     ; INPUT ->
1590                              <1>     ;       ESI = 11 byte DOS File Name location
1591                              <1>     ;       (in DOS Directory Entry Format)
1592                              <1>     ; OUTPUT ->
1593                              <1>     ;        AL = 8 bit checksum (CRC) value
```

```
1594                                    <1>     ;
1595                                    <1>     ; (Modified registers: EAX, ECX, ESI)
1596                                    <1>
1597                                    <1>     ; Erdogan Tan [ 17-10-2009 ]
1598                                    <1>     ;  'ror al, 1' instruction
1599                                    <1>
1600                                    <1>     ; Erdogan Tan [ 20-06-2004 ]
1601                                    <1>     ; This 8086 assembly code is an original code
1602                                    <1>     ; which is adapted from C code in
1603                                    <1>     ; Microsoft FAT32 File System Specification
1604                                    <1>     ; Version 1.03, December 6, 2000
1605                                    <1>     ; Page 28
1606                                    <1>
1607 0000AA3F 30C0                     <1>     xor     al, al
1608                                    <1>     ;mov    ecx, 11
1609                                    <1>     ; 29/07/2022
1610 0000AA41 29C9                     <1>     sub     ecx, ecx
1611 0000AA43 B10B                     <1>     mov     cl, 11
1612                                    <1> loc_next_sum:
1613                                    <1>     ;xor    ah, ah
1614                                    <1>     ;test   al, 1
1615                                    <1>     ;jz     short pass_ah_80h
1616                                    <1>     ;mov    ah, 80h
1617                                    <1> ;pass_ah_80h:
1618                                    <1>     ;shr    al, 1
1619 0000AA45 D0C8                     <1>     ror     al, 1 ; 17/10/2009
1620 0000AA47 0206                     <1>     add     al, [esi]
1621 0000AA49 46                       <1>     inc     esi
1622                                    <1>     ;add    al, ah
1623 0000AA4A E2F9                     <1>     loop    loc_next_sum
1624 0000AA4C C3                       <1>     retn
1625                                    <1>
1626                                    <1> get_fs_longname:
1627                                    <1>     ; temporary (13/02/2016)
1628 0000AA4D 31C0                     <1>     xor     eax, eax
1629 0000AA4F F9                       <1>     stc
1630 0000AA50 C3                       <1>     retn
1631                                    <1>
1632                                    <1> make_sub_directory:
1633                                    <1>     ; 19/12/2025 (TRDOS 386 Kernel v2.0.10)
1634                                    <1>     ; 07/08/2022
1635                                    <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
1636                                    <1>     ; 16/10/2016
1637                                    <1>     ; 02/03/2016, 03/03/2016
1638                                    <1>     ; 26/02/2016, 27/02/2016
1639                                    <1>     ; 21/02/2016 (TRDOS 386 = TRDOS v2.0)
1640                                    <1>     ; 01/08/2011 (DIR.ASM, 'proc_make_directory')
1641                                    <1>     ; 10/07/2010
1642                                    <1>     ; INPUT ->
1643                                    <1>     ;       ESI = ASCIIZ Directory Name
1644                                    <1>     ;       CL = Directory Attributes
1645                                    <1>     ; OUTPUT ->
1646                                    <1>     ;       EAX = New sub dir's first cluster
1647                                    <1>     ;       ESI = Logical Dos Drv Descr. Table Addr.
1648                                    <1>     ;       CF = 1 -> error code in AL (EAX)
1649                                    <1>
1650                                    <1>     ;test   cl, 10h  ; directory
1651                                    <1>     ;jz     short loc_make_directory_access_denied
1652                                    <1>     ;test   cl, 08h ; volume name
1653                                    <1>     ;jnz    short loc_make_directory_access_denied
1654                                    <1>
1655 0000AA51 80E107                   <1>     and     cl, 07h
1656 0000AA54 880D[C0800100]           <1>     mov     byte [mkdir_attrib], cl
1657                                    <1>
1658 0000AA5A 56                       <1>     push    esi
1659 0000AA5B 31DB                     <1>     xor     ebx, ebx
1660 0000AA5D 8A3D[42770100]           <1>     mov     bh, [Current_Drv]
1661 0000AA63 BE00010900               <1>     mov     esi, Logical_DOSDisks
1662 0000AA68 01DE                     <1>     add     esi, ebx
1663 0000AA6A 5B                       <1>     pop     ebx
1664                                    <1>
1665                                    <1>     ; 10/07/2010 -> 1st writable disk check for trdos
1666                                    <1>     ; LD_DiskType = 0 for write protection (read only)
1667 0000AA6B 807E0101                 <1>     cmp     byte [esi+LD_DiskType], 1 ; 0 = Invalid
1668 0000AA6F 7308                     <1>     jnb     short loc_mkdir_check_file_sytem
1669                                    <1>     ; 16/10/2016 (13h -> 30)
1670                                    <1>     ;mov    eax, 30 ; 'Disk write-protected' error
1671                                    <1>     ;mov    edx, 0
1672                                    <1>     ; 29/07/2022
1673 0000AA71 29C0                     <1>     sub     eax, eax
1674 0000AA73 B01E                     <1>     mov     al, 30
1675 0000AA75 29D2                     <1>     sub     edx, edx ; 0
1676 0000AA77 F9                       <1>     stc
1677                                    <1>     ; err retn: EDX = 0, EBX = Dir name offset
1678                                    <1>     ;ESI = Logical DOS drive description table address
1679 0000AA78 C3                       <1>     retn
1680                                    <1>
1681                                    <1> ;loc_make_directory_access_denied:
1682                                    <1>     ;mov    ax, 05h ; access denied (invalid attributes input)
1683                                    <1>     ;stc
1684                                    <1>     ;retn
1685                                    <1>
1686                                    <1> loc_mkdir_check_file_sytem:
1687                                    <1> ; 19/12/2025
1688                                    <1> %if 0
1689                                    <1>     cmp     byte [esi+LD_FATType], 1
1690                                    <1>     jnb     short loc_mkdir_check_free_sectors
1691                                    <1>
1692                                    <1> loc_make_fs_directory:
1693                                    <1>     mov     eax, [Current_Dir_FCluster]
1694                                    <1>
1695                                    <1>     ; EAX = Parent directory DDT Address
1696                                    <1>     ; ESI = Logical DOS Drive DT Address
1697                                    <1>     ; EBX = Directory name offset (as ASCIIZ name)
1698                                    <1>
1699                                    <1>     ;call   make_fs_directory
1700                                    <1>     ;retn
1701                                    <1>     ; 29/07/2022
1702                                    <1>     jmp     make_fs_directory
1703                                    <1> %endif
1704                                    <1>
1705                                    <1> loc_mkdir_check_free_sectors:
1706 0000AA79 0FB64613                 <1>         movzx   eax, byte [esi+LD_BPB+SecPerClust]
1707 0000AA7D 8B4E74                   <1>     mov     ecx, [esi+LD_FreeSectors]
1708 0000AA80 39C1                     <1>     cmp     ecx, eax
1709 0000AA82 7254                     <1>     jb      short loc_mkdir_insufficient_disk_space
1710                                    <1>
1711                                    <1> loc_make_fat_directory:
1712 0000AA84 891D[B0800100]           <1>     mov     [mkdir_DirName_Offset], ebx
1713 0000AA8A 890D[BC800100]           <1>     mov     [mkdir_FreeSectors], ecx
1714                                    <1>
1715                                    <1>     ;mov    al, [esi+LD_BPB+SecPerClust]
1716 0000AA90 A2[C2800100]             <1>     mov     byte [mkdir_SecPerClust], al
1717                                    <1>
```

```
1718                                    <1> loc_mkdir_gffc_1:
1719 0000AA95 E8F1160000                <1>     call    get_first_free_cluster
1720 0000AA9A 722A                      <1>     jc      short loc_mkdir_gffc_retn
1721                                    <1>
1722                                    <1> ;loc_mkdir_gffc_1_cont:
1723                                    <1>     ;cmp    eax, 2
1724                                    <1>     ;jb     short loc_mkdir_gffc_insufficient_disk_space
1725                                    <1>
1726                                    <1> ;loc_mkdir_gffc_1_save_fcluster:
1727 0000AA9C A3[B4800100]              <1>     mov     [mkdir_FFCluster], eax
1728                                    <1>
1729                                    <1> loc_mkdir_locate_ffe:
1730                                    <1>     ; Current directory fcluster <> Directory buffer cluster
1731                                    <1>     ; Current directory will be reloaded by
1732                                    <1>     ; 'locate_current_dir_file' procedure
1733                                    <1>     ;
1734                                    <1>     ; ESI = Logical DOS Drive Description Table Address
1735                                    <1>     ;push   esi ; 27/02/2016
1736 0000AAA1 31C0                      <1>     xor     eax, eax
1737 0000AAA3 89C1                      <1>         mov ecx, eax
1738 0000AAA5 6649                      <1>     dec     cx ; FFFFh
1739                                    <1>     ; CX = FFFFh -> find first deleted or free entry
1740                                    <1>     ; ESI would be ASCIIZ filename address if the call
1741                                    <1>     ; would not be for first free or deleted dir entry
1742 0000AAA7 E84CFBFFFF                <1>     call    locate_current_dir_file
1743 0000AAAC 734A                      <1>     jnc     short loc_mkdir_set_ff_dir_entry_1
1744                                    <1>     ;pop    esi
1745                                    <1>     ; ESI = Logical DOS Drive Description Table Address
1746 0000AAAE 83F802                    <1>     cmp     eax, 2  ; cmp al, 2 ; File/Dir not found !
1747 0000AAB1 7529                      <1>     jne     short loc_mkdir_stc_return
1748                                    <1>
1749                                    <1> loc_mkdir_add_new_cluster:
1750 0000AAB3 3805[41770100]            <1>     cmp     byte [Current_FATType], al ; 2
1751                                    <1>     ;cmp    byte ptr [esi+LD_FATType], 2
1752 0000AAB9 770C                      <1>     ja      short loc_mkdir_add_new_cluster_check_fsc
1753 0000AABB 803D[40770100]01          <1>     cmp     byte [Current_Dir_Level], 1
1754                                    <1>     ;cmp    byte [esi+LD_CDirLevel], 1
1755 0000AAC2 7303                      <1>     jnb     short loc_mkdir_add_new_cluster_check_fsc
1756                                    <1>
1757 0000AAC4 B00C                      <1>     mov     al, 12 ; No more files
1758                                    <1> loc_mkdir_gffc_retn:
1759 0000AAC6 C3                        <1>     retn
1760                                    <1>
1761                                    <1> loc_mkdir_add_new_cluster_check_fsc:
1762 0000AAC7 8B0D[BC800100]            <1>     mov     ecx, [mkdir_FreeSectors]
1763                                    <1>     ;movzx  eax, byte [mkdir_SecPerClust]
1764 0000AACD A0[C2800100]              <1>     mov     al, [mkdir_SecPerClust]
1765                                    <1>     ;shl    ax, 1 ; AX = 2 * AX
1766                                    <1>     ; 29/07/2022
1767 0000AAD2 D1E0                      <1>     shl     eax, 1
1768 0000AAD4 39C1                      <1>     cmp     ecx, eax
1769 0000AAD6 7350                      <1>     jnb     short loc_mkdir_add_new_subdir_cluster
1770                                    <1>
1771                                    <1> loc_mkdir_insufficient_disk_space:
1772                                    <1>     ;mov    edx, ecx
1773                                    <1> ;loc_mkdir_gffc_insufficient_disk_space:
1774                                    <1>     ; 29/07/2022
1775                                    <1>     ;mov    ax, 27h ; MSDOS err => insufficient disk space
1776                                    <1>
1777                                    <1>     ; err retn: EDX = Free sectors, EBX = Dir name offset
1778                                    <1>         ; ESI -> Dos drive description table address
1779                                    <1>     ;; ecx = edx
1780                                    <1>
1781                                    <1>     ;retn
1782                                    <1>
1783                                    <1>     ; 29/07/2022
1784 0000AAD8 30E4                      <1>     xor     ah, ah
1785 0000AADA B027                      <1>     mov     al, 27h
1786                                    <1>
1787                                    <1> loc_mkdir_stc_return:
1788 0000AADC F9                        <1>     stc
1789 0000AADD C3                        <1>     retn
1790                                    <1>
1791                                    <1> loc_mkdir_gffc_2:
1792 0000AADE E8A8160000                <1>     call    get_first_free_cluster
1793 0000AAE3 72E1                      <1>     jc      short loc_mkdir_gffc_retn
1794                                    <1>
1795                                    <1> ;loc_mkdir_gffc_1_cont:
1796                                    <1>     ;cmp    eax, 2
1797                                    <1>     ;jb     short loc_mkdir_gffc_insufficient_disk_space
1798                                    <1>
1799                                    <1> ;loc_mkdir_gffc_2_save_fcluster:
1800 0000AAE5 A3[B4800100]              <1>     mov     [mkdir_FFCluster], eax
1801                                    <1>
1802 0000AAEA A1[B8800100]              <1>     mov     eax, [mkdir_LastDirCluster]
1803                                    <1>
1804 0000AAEF E841160000                <1>     call    load_FAT_sub_directory
1805 0000AAF4 72D0                      <1>     jc      short loc_mkdir_gffc_retn
1806                                    <1>
1807 0000AAF6 31FF                      <1>     xor     edi, edi
1808                                    <1> loc_mkdir_set_ff_dir_entry_1:
1809                                    <1>     ; 27/02/2016
1810 0000AAF8 56                        <1>     push    esi ; Logical DOS Drv Desc. Tbl. address
1811                                    <1>     ; EDI = Directory Entry Address
1812 0000AAF9 8B35[B0800100]            <1>     mov     esi, [mkdir_DirName_Offset]
1813 0000AAFF A1[B4800100]              <1>     mov     eax, [mkdir_FFCluster]
1814                                    <1>
1815 0000AB04 66B91000                  <1>     mov     cx, 10h ; CL = Directory attribute
1816                                    <1>                     ; CH = 0 -> File size is 0
1817 0000AB08 0A0D[C0800100]            <1>     or      cl, [mkdir_attrib] ; S, H, R
1818 0000AB0E E89B010000                <1>     call    make_directory_entry
1819                                    <1>
1820 0000AB13 5E                        <1>     pop     esi
1821                                    <1>
1822 0000AB14 C605[697E0100]02          <1>     mov     byte [DirBuff_ValidData], 2
1823 0000AB1B E861020000                <1>     call    save_directory_buffer
1824                                    <1>         ;jnc       loc_mkdir_set_ff_dir_entry_2
1825                                    <1>     ; 29/07/2022
1826 0000AB20 7205                      <1>     jc      short loc_mkdir_return
1827 0000AB22 E9CA000000                <1>     jmp     loc_mkdir_set_ff_dir_entry_2
1828                                    <1>
1829                                    <1> loc_mkdir_return:
1830 0000AB27 C3                        <1>     retn
1831                                    <1>
1832                                    <1> loc_mkdir_add_new_subdir_cluster:
1833 0000AB28 8B15[6E7E0100]            <1>     mov     edx, [DirBuff_Cluster]
1834 0000AB2E 8915[B8800100]            <1>     mov     [mkdir_LastDirCluster], edx
1835                                    <1>
1836 0000AB34 A1[B4800100]              <1>     mov     eax, [mkdir_FFCluster]
1837 0000AB39 E8F7150000                <1>     call    load_FAT_sub_directory
1838 0000AB3E 72E7                      <1>     jc      short loc_mkdir_return
1839                                    <1>     ; eax = 0
1840                                    <1>     ; ecx = directory buffer sector count (<= 128)
1841                                    <1>
```

```
1842                                <1> pass_mkdir_add_new_subdir_cluster:
1843                                <1>
1844                                <1> ; 29/07/2022
1845                                <1> ;   ;sub    edi, edi ; 0
1846                                <1> ;   ; 29/07/2022 - BUGFIX !
1847                                <1> ;   mov     edi, Directory_Buffer
1848                                <1> ;
1849                                <1> ;   ;mov    al, 128 ; double word
1850                                <1> ;   ;mul    ecx ; ecx =  directory buffer sector count
1851                                <1> ;   ;mov    ecx, eax
1852                                <1> ;   ;shl    cx, 7 ; 128 * sector count
1853                                <1> ;   mov     ax, [esi+LD_BPB+BytesPerSec] ; 512
1854                                <1> ;   ;shr    ax, 2 ; 'byte count / 4' for 'stosd'
1855                                <1> ;   ; 29/07/2022
1856                                <1> ;   shr     eax, 2
1857                                <1> ;   ;mul    cx ; max = 128*(512/4) -> 16384 (stosd)
1858                                <1> ;   ;mov    cx, ax
1859                                <1> ;   ;sub    ax, ax ; 0
1860                                <1> ;   ; 29/07/2022
1861                                <1> ;   mul     ecx
1862                                <1> ;   mov     ecx, eax
1863                                <1> ;   sub     eax, eax
1864                                <1> ;   rep     stosd ; clear directory buffer
1865                                <1>
1866                                <1>     ; 29/07/2022
1867 0000AB40 E85C010000           <1>     call    clear_directory_buffer
1868                                <1>
1869 0000AB45 C605[697E0100]02     <1>     mov     byte [DirBuff_ValidData], 2
1870 0000AB4C E830020000           <1>     call    save_directory_buffer
1871 0000AB51 72D4                 <1>     jc      short loc_mkdir_return
1872                                <1>
1873                                <1> loc_mkdir_save_added_cluster:
1874 0000AB53 A1[B8800100]         <1>     mov     eax, [mkdir_LastDirCluster]
1875 0000AB58 8B0D[B4800100]       <1>     mov     ecx, [mkdir_FFCluster]
1876                                <1>     ; 01/03/2016
1877 0000AB5E 31D2                 <1>     xor     edx, edx
1878 0000AB60 8915[5E7E0100]       <1>     mov     [FAT_ClusterCounter], edx ; 0 ; reset
1879 0000AB66 E8F5160000           <1>     call    update_cluster
1880 0000AB6B 7304                 <1>     jnc     short loc_mkdir_save_fat_buffer_0
1881 0000AB6D 09C0                 <1>     or      eax, eax ; EAX = 0 -> cluster value is 0 or eocc
1882 0000AB6F 7518                 <1>     jnz     short loc_mkdir_save_fat_buffer_stc_retn
1883                                <1>
1884                                <1> loc_mkdir_save_fat_buffer_0:
1885 0000AB71 A1[B4800100]         <1>     mov     eax, [mkdir_FFCluster]
1886 0000AB76 A3[B8800100]         <1>     mov     [mkdir_LastDirCluster], eax
1887                                <1>
1888 0000AB7B 31C9                 <1>     xor     ecx, ecx
1889 0000AB7D 49                   <1>     dec     ecx ; FFFFFFFFh
1890                                <1>     ; ESI = Logical DOS Drive Description Table address
1891 0000AB7E E8DD160000           <1>     call    update_cluster
1892 0000AB83 731A                 <1>     jnc     short loc_mkdir_save_fat_buffer_1
1893 0000AB85 09C0                 <1>     or      eax, eax
1894 0000AB87 7416                 <1>     jz      short loc_mkdir_save_fat_buffer_1
1895                                <1>
1896                                <1> loc_mkdir_save_fat_buffer_stc_retn:
1897                                <1>     ; 01/03/2016
1898 0000AB89 803D[5E7E0100]01     <1>     cmp     byte [FAT_ClusterCounter], 1
1899 0000AB90 720C                 <1>     jb      short loc_mkdir_save_fat_buffer_retn
1900                                <1>
1901 0000AB92 66BB00FF             <1>     mov     bx, 0FF00h ; recalculate free space (BL = 0)
1902                                <1>                        ; (BH = FFh -> Use ESI as Drv Param. Tbl.)
1903 0000AB96 50                   <1>     push    eax
1904 0000AB97 E8C5190000           <1>     call    calculate_fat_freespace
1905 0000AB9C 58                   <1>     pop     eax
1906 0000AB9D F9                   <1>     stc
1907                                <1> loc_mkdir_save_fat_buffer_retn:
1908 0000AB9E C3                   <1>     retn
1909                                <1>
1910                                <1> loc_mkdir_save_fat_buffer_1:
1911                                <1>     ; byte [FAT_BuffValidData] = 2
1912 0000AB9F E830190000           <1>     call    save_fat_buffer
1913 0000ABA4 72E3                 <1>     jc      short loc_mkdir_save_fat_buffer_stc_retn
1914                                <1>
1915                                <1>     ; 01/03/2016
1916 0000ABA6 803D[5E7E0100]01     <1>     cmp     byte [FAT_ClusterCounter], 1
1917 0000ABAD 721B                 <1>     jb      short loc_mkdir_save_fat_buffer_2
1918                                <1>
1919                                <1>     ; ESI = Logical DOS Drive Description Table address
1920 0000ABAF A1[5E7E0100]         <1>     mov     eax, [FAT_ClusterCounter]
1921 0000ABB4 66BB01FF             <1>     mov     bx, 0FF01h ; add free clusters
1922 0000ABB8 E8A4190000           <1>     call    calculate_fat_freespace
1923                                <1>
1924                                <1>     ;inc    eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
1925                                <1>     ;jnz    short loc_mkdir_save_fat_buffer_2
1926                                <1>
1927                                <1>     ; ecx > 0 -> Recalculation is needed
1928 0000ABBD 09C9                 <1>     or      ecx, ecx
1929 0000ABBF 7409                 <1>     jz      short loc_mkdir_save_fat_buffer_2
1930                                <1>
1931 0000ABC1 66BB00FF             <1>     mov     bx, 0FF00h ; ; recalculate free space
1932 0000ABC5 E897190000           <1>     call    calculate_fat_freespace
1933                                <1>
1934                                <1> loc_mkdir_save_fat_buffer_2:
1935 0000ABCA C605[C3800100]01     <1>     mov     byte [mkdir_add_new_cluster], 1
1936 0000ABD1 E9B0000000           <1>     jmp     loc_mkdir_upd_parent_dir_lmdt
1937                                <1>
1938                                <1> loc_mkdir_update_sub_dir_cluster:
1939 0000ABD6 A1[B4800100]         <1>     mov     eax, [mkdir_FFCluster]
1940 0000ABDB 29C9                 <1>     sub     ecx, ecx ; 0
1941                                <1>     ; 01/03/2016
1942 0000ABDD 890D[5E7E0100]       <1>     mov     [FAT_ClusterCounter], ecx ; 0 ; Reset
1943 0000ABE3 49                   <1>     dec     ecx ; 0FFFFFFFFh
1944                                <1>
1945                                <1>     ; ESI = Logical DOS Drive Description Table address
1946 0000ABE4 E877160000           <1>     call    update_cluster
1947 0000ABE9 7364                 <1>     jnc     short loc_mkdir_save_fat_buffer_3
1948 0000ABEB 09C0                 <1>     or      eax, eax ; EAX = 0 -> cluster value is 0 or eocc
1949 0000ABED 7460                 <1>     jz      short loc_mkdir_save_fat_buffer_3
1950                                <1>     ; 01/03/2016
1951 0000ABEF EB98                 <1>     jmp     short loc_mkdir_save_fat_buffer_stc_retn
1952                                <1>
1953                                <1> loc_mkdir_set_ff_dir_entry_2:
1954                                <1>     ; ESI = Logical DOS Drive Description Table address
1955 0000ABF1 A1[B4800100]         <1>     mov     eax, [mkdir_FFCluster]
1956                                <1>     ; Load disk sectors as a directory cluster
1957 0000ABF6 E83A150000           <1>     call    load_FAT_sub_directory
1958 0000ABFB 7251                 <1>     jc      short retn_make_fat_directory
1959                                <1>
1960                                <1>     ; eax = 0
1961                                <1>     ; ecx = directory buffer sector count (<= 128)
1962                                <1>
1963                                <1> ; 29/07/2022
1964                                <1> ;   mov     edi, Directory_Buffer + 64 ; 26/02/2016
1965                                <1> ;
```

```
1966                                    <1> ;   ; 02/03/2016
1967                                    <1> ;   mov     ax, [esi+LD_BPB+BytesPerSec] ; 512
1968                                    <1> ;   ;shr    ax, 2 ; 'byte count / 4' for 'stosd'
1969                                    <1> ;   ; 29/07/2022
1970                                    <1> ;   shr     eax, 2
1971                                    <1> ;   mul     ecx
1972                                    <1> ;   mov     ecx, eax
1973                                    <1> ;   ;
1974                                    <1> ;   ; 29/07/2022 - BUGFIX !
1975                                    <1> ;   sub     ecx, 16 ; - 64 bytes
1976                                    <1> ;                   ; (space for '.' & '..' entries)
1977                                    <1> ;   ;sub    ax, ax
1978                                    <1> ;   sub     eax, eax
1979                                    <1> ;   rep     stosd
1980                                    <1> ;
1981                                    <1> ;   ;;mov   al, 128 ; double word (count in sector)
1982                                    <1> ;   ;;mul   ecx ; ecx = directory buffer sector count
1983                                    <1> ;   ;;mov   ecx, eax
1984                                    <1> ;   ;shl    cx, 7 ; 128 * sector count
1985                                    <1> ;   ;sub    ecx, 64 ; 29/07/2022
1986                                    <1> ;   ;;sub   eax, eax
1987                                    <1> ;   ;;sub   al, al ; 0
1988                                    <1> ;   ;rep    stosd ; clear directory buffer
1989                                    <1>
1990                                    <1>     ; 29/07/2022
1991 0000ABFD E89F000000                <1>     call    clear_directory_buffer
1992                                    <1>
1993 0000AC02 BF00000800                <1>     mov     edi, Directory_Buffer ; 26/02/2016
1994                                    <1>
1995 0000AC07 56                        <1>     push    esi
1996                                    <1>
1997 0000AC08 BE[C4800100]              <1>     mov     esi, mkdir_Name
1998 0000AC0D 66C7062E00                <1>     mov     word [esi], 2Eh ; db '.', '0'
1999                                    <1>
2000 0000AC12 A1[B4800100]              <1>     mov     eax, [mkdir_FFCluster]
2001                                    <1>     ;mov    cx, 10h ; CL = Directory attribute
2002                                    <1>                     ; CH = 0 -> File size is 0
2003                                    <1>     ; 29/07/2022
2004 0000AC17 B110                      <1>     mov     cl, 10h
2005 0000AC19 E890000000                <1>     call    make_directory_entry
2006                                    <1>
2007 0000AC1E BF20000800                <1>     mov     edi, Directory_Buffer + 32 ; 26/02/2016
2008                                    <1>
2009                                    <1>     ; 03/03/2016
2010                                    <1>     ; Following modification has been done according to
2011                                    <1>     ; 'Microsoft Extensible Firmware Initiative
2012                                    <1>     ; FAT32 File System Specification' document,
2013                                    <1>     ; 'FAT: General Overview of On-Disk Format-Page 25'.
2014                                    <1>     ; "Finally, you set DIR_FstClusLO and DIR_FstClusHI
2015                                    <1>     ; for the dotdot entry (the second entry) to the
2016                                    <1>     ; first cluster number of the directory in which you
2017                                    <1>     ; just created the directory (value is 0 if this directory
2018                                    <1>     ; is the root directory even for FAT32 volumes)."
2019                                    <1>     ; (Correctness of this modification has been verified
2020                                    <1>     ;  by using Windows 98 'scandisk.exe'.)
2021                                    <1>
2022 0000AC23 29C0                      <1>     sub     eax, eax
2023 0000AC25 3805[40770100]            <1>     cmp     byte [Current_Dir_Level], al ; 0
2024 0000AC2B 7605                      <1>     jna     short loc_mkdir_set_ff_dir_entry_3
2025 0000AC2D A1[3C770100]              <1>     mov     eax, [Current_Dir_FCluster] ; parent dir
2026                                    <1> loc_mkdir_set_ff_dir_entry_3:
2027 0000AC32 66C746012E00              <1>     mov     word [esi+1], 2Eh ; db '.', '0'
2028                                    <1>
2029                                    <1>     ;mov    cx, 10h
2030 0000AC38 E871000000                <1>     call    make_directory_entry
2031                                    <1>
2032 0000AC3D 5E                        <1>     pop     esi
2033                                    <1>
2034 0000AC3E C605[697E0100]02          <1>     mov     byte [DirBuff_ValidData], 2
2035 0000AC45 E837010000                <1>     call    save_directory_buffer
2036                                    <1>     ;jnc    loc_mkdir_update_sub_dir_cluster
2037                                    <1>     ; 29/07/2022
2038 0000AC4A 7202                      <1>     jc      short retn_make_fat_directory
2039 0000AC4C EB88                      <1>     jmp     loc_mkdir_update_sub_dir_cluster
2040                                    <1>
2041                                    <1> retn_make_fat_directory:
2042 0000AC4E C3                        <1>     retn
2043                                    <1>
2044                                    <1> loc_mkdir_save_fat_buffer_3:
2045                                    <1>     ; 01/03/2016
2046                                    <1>     ; byte [FAT_BuffValidData] = 2
2047 0000AC4F E880180000                <1>     call    save_fat_buffer
2048                                    <1>     ;jc     short loc_mkdir_save_fat_buffer_stc_retn
2049                                    <1>     ; 07/08/2022
2050 0000AC54 7305                      <1>     jnc     short loc_mkdir_save_fat_buffer_4
2051 0000AC56 E92EFFFFFF                <1>     jmp     loc_mkdir_save_fat_buffer_stc_retn
2052                                    <1>
2053                                    <1> loc_mkdir_save_fat_buffer_4:
2054 0000AC5B 803D[5E7E0100]01          <1>     cmp     byte [FAT_ClusterCounter], 1
2055 0000AC62 721B                      <1>     jb      short loc_mkdir_save_fat_buffer_5
2056                                    <1>
2057                                    <1>     ; ESI = Logical DOS Drive Description Table address
2058 0000AC64 A1[5E7E0100]              <1>     mov     eax, [FAT_ClusterCounter]
2059 0000AC69 66BB01FF                  <1>     mov     bx, 0FF01h ; add free clusters
2060 0000AC6D E8EF180000                <1>     call    calculate_fat_freespace
2061                                    <1>
2062                                    <1>     ;inc    eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
2063                                    <1>         ;jnz    short loc_mkdir_save_fat_buffer_5
2064                                    <1>
2065                                    <1>     ; ecx > 0 -> Recalculation is needed
2066 0000AC72 09C9                      <1>     or      ecx, ecx
2067 0000AC74 7409                      <1>         jz      short loc_mkdir_save_fat_buffer_5
2068                                    <1>
2069 0000AC76 66BB00FF                  <1>     mov     bx, 0FF00h ; recalculate free space
2070 0000AC7A E8E2180000                <1>     call    calculate_fat_freespace
2071                                    <1>
2072                                    <1> loc_mkdir_save_fat_buffer_5:
2073 0000AC7F C605[C3800100]00          <1>     mov     byte [mkdir_add_new_cluster], 0
2074                                    <1>
2075                                    <1> loc_mkdir_upd_parent_dir_lmdt:
2076 0000AC86 E87F010000                <1>     call    update_parent_dir_lmdt
2077                                    <1>
2078                                    <1>     ; 01/03/2016
2079 0000AC8B 803D[C3800100]00          <1>     cmp     byte [mkdir_add_new_cluster], 0
2080                                    <1>     ;ja     loc_mkdir_gffc_2
2081                                    <1>     ; 29/07/2022
2082 0000AC92 7605                      <1>     jna     short loc_mkdir_retn_new_dir_cluster
2083 0000AC94 E945FEFFFF                <1>     jmp     loc_mkdir_gffc_2
2084                                    <1>
2085                                    <1> loc_mkdir_retn_new_dir_cluster:
2086 0000AC99 A1[B4800100]              <1>     mov     eax, [mkdir_FFCluster]
2087 0000AC9E 31D2                      <1>     xor     edx, edx
2088                                    <1> loc_mkdir_retn:
2089 0000ACA0 C3                        <1>     retn
```

```
                                <1>
                                <1> clear_directory_buffer:
                                <1>     ; 19/12/2025 (TRDOS 386 Kernel v2.0.10)
                                <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <1>     ;
                                <1>     ; eax = 0
                                <1>     ; ecx = directory buffer sector count (<= 128)
                                <1>     ;
0000ACA1 BF00000800             <1>     mov     edi, Directory_Buffer
                                <1>     ;mov    al, 128 ; double word
                                <1>     ;mul    ecx ; ecx = directory buffer sector count
                                <1>     ;mov    ecx, eax
                                <1>     ;shl    cx, 7 ; 128 * sector count
                                <1>     ; 19/12/2025
                                <1>     ;mov    ax, [esi+LD_BPB+BytesPerSec] ; 512
                                <1>     ;shr    eax, 2 ; 'byte count / 4' for 'stosd'
                                <1>     ;mov    ax, 128 ; 512/4
                                <1>     ;mul    ecx ; max = 128*(512/4) -> 16384 (stosd)
                                <1>     ;mov    ecx, eax
0000ACA6 C1E107                 <1>     shl     ecx, 7 ; (512/4) * sector count
0000ACA9 29C0                   <1>     sub     eax, eax
0000ACAB F3AB                   <1>     rep     stosd ; clear directory buffer
0000ACAD C3                     <1>     retn
                                <1>
                                <1> make_directory_entry:
                                <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <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>
0000ACAE 51                     <1>     push    ecx
                                <1>
0000ACAF 884F0B                 <1>     mov     [edi+11], cl ; Attributes
0000ACB2 6689471A               <1>     mov     [edi+26], ax ; FClusterLw, 26
0000ACB6 C1E810                 <1>     shr     eax, 16
0000ACB9 66894714               <1>     mov     [edi+20], ax ; FClusterHw, 20
                                <1>     ;xor    ax, ax
                                <1>     ; 29/07/2022
0000ACBD 31C0                   <1>     xor     eax, eax
0000ACBF 6689470C               <1>     mov     [edi+12], ax ; NTReserved, 12
                                <1>                          ; CrtTimeTenth, 13
0000ACC3 08ED                   <1>     or      ch, ch
0000ACC5 7402                   <1>     jz      short loc_make_direntry_set_filesize
                                <1>
0000ACC7 8B03                   <1>     mov     eax, [ebx]
                                <1>
                                <1> loc_make_direntry_set_filesize:
0000ACC9 89471C                 <1>     mov     [edi+28], eax ; FileSize, 28
                                <1>
0000ACCC E8D3FBFFFF             <1>     call    convert_file_name
                                <1>     ; EDI = Dir Entry Format File Name Location
                                <1>     ; ESI = Dot File Name Location (capitalized)
                                <1>
0000ACD1 E816000000             <1>     call    convert_current_date_time
                                <1>     ; OUTPUT -> DX = Date in dos dir entry format
                                <1>         ;       AX = Time in dos dir entry format
0000ACD6 6689470E               <1>     mov     [edi+14], ax ; CrtTime, 14
0000ACDA 66895710               <1>     mov     [edi+16], dx ; CrtDate, 16
0000ACDE 66895712               <1>     mov     [edi+18], dx ; LastAccDate, 18
0000ACE2 66894716               <1>     mov     [edi+22], ax ; WrtTime, 14
0000ACE6 66895718               <1>     mov     [edi+24], dx ; WrtDate, 16
0000ACEA 59                     <1>     pop     ecx
                                <1>
0000ACEB C3                     <1>     retn
                                <1>
                                <1> convert_current_date_time:
                                <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <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>
0000ACEC B404                   <1>     mov     ah, 04h ; Return Current Date
0000ACEE E8EAB5FFFF             <1>     call    int1Ah
                                <1>
0000ACF3 88E8                   <1>     mov     al, ch ; <- century BCD
0000ACF5 240F                   <1>     and     al, 0Fh
0000ACF7 88EC                   <1>     mov     ah, ch
0000ACF9 C0EC04                 <1>     shr     ah, 4
0000ACFC D50A                   <1>     aad
0000ACFE 88C5                   <1>     mov     ch, al ; -> century
                                <1>
0000AD00 88C8                   <1>     mov     al, cl ; <- year BCD
0000AD02 240F                   <1>     and     al, 0Fh
0000AD04 88CC                   <1>     mov     ah, cl
0000AD06 C0EC04                 <1>     shr     ah, 4
0000AD09 D50A                   <1>     aad
0000AD0B 88C1                   <1>     mov     cl, al ; -> year
                                <1>
                                <1>     ;mov    al, ch
                                <1>     ;mov    ah, 100
                                <1>     ;mul    ah
                                <1>     ; 29/07/2022
0000AD0D B064                   <1>     mov     al, 100
0000AD0F F6E5                   <1>     mul     ch
                                <1>
                                <1>     ;xor    ch, ch
                                <1>     ;add    ax, cx
                                <1>     ; 29/07/2022
0000AD11 00C8                   <1>     add     al, cl
0000AD13 80D400                 <1>     adc     ah, 0
0000AD16 662DBC07               <1>     sub     ax, 1980 ; ms-dos epoch
                                <1>     ;mov    cx, ax
                                <1>     ; 29/07/2022
0000AD1A 88C1                   <1>     mov     cl, al
```

```
                                <1>     ;mov    ecx, eax
                                <1>
0000AD1C 88F0                   <1>     mov     al, dh ; <- month in bcd
0000AD1E 240F                   <1>     and     al, 0Fh
0000AD20 88F4                   <1>     mov     ah, dh
0000AD22 C0EC04                 <1>     shr     ah, 4
0000AD25 D50A                   <1>     aad
0000AD27 88C6                   <1>     mov     dh, al ; -> month
                                <1>
0000AD29 88D0                   <1>     mov     al, dl ; <- day BCD
0000AD2B 240F                   <1>     and     al, 0Fh
0000AD2D 88D4                   <1>     mov     ah, dl
0000AD2F C0EC04                 <1>     shr     ah, 4
0000AD32 D50A                   <1>     aad
0000AD34 88C2                   <1>     mov     dl, al ; -> day
                                <1>
0000AD36 88C8                   <1>     mov     al, cl ; count of years from 1980
                                <1>     ;shl    ax, 4
                                <1>     ; 29/07/2022
                                <1>     ;mov    eax, ecx
0000AD38 C1E004                 <1>     shl     eax, 4
                                <1>
0000AD3B 08F0                   <1>     or      al, dh ; month of year, 1 to 12
                                <1>     ;shl    ax, 5
                                <1>     ; 29/07/2022
0000AD3D C1E005                 <1>     shl     eax, 5
0000AD40 08D0                   <1>     or      al, dl ; day of year, 1 to 31
                                <1>
                                <1>     ;push   ax ; push date
                                <1>     ; 29/07/2022
0000AD42 50                     <1>     push    eax
                                <1>
0000AD43 B402                   <1>     mov     ah, 02h ; Return Current Time
0000AD45 E893B5FFFF             <1>     call    int1Ah
                                <1>
0000AD4A 88E8                   <1>     mov     al, ch ; <- hours BCD
0000AD4C 240F                   <1>     and     al, 0Fh
0000AD4E 88EC                   <1>     mov     ah, ch
0000AD50 C0EC04                 <1>     shr     ah, 4
0000AD53 D50A                   <1>     aad
0000AD55 88C5                   <1>     mov     ch, al ; -> hours
                                <1>
0000AD57 88C8                   <1>     mov     al, cl ; <- minutes BCD
0000AD59 240F                   <1>     and     al, 0Fh
0000AD5B 88CC                   <1>     mov     ah, cl
0000AD5D C0EC04                 <1>     shr     ah, 4
0000AD60 D50A                   <1>     aad
0000AD62 88C1                   <1>     mov     cl, al ; -> minutes
                                <1>
0000AD64 88F0                   <1>     mov     al, dh ; <- seconds BCD
0000AD66 240F                   <1>     and     al, 0Fh
0000AD68 88F4                   <1>     mov     ah, dh
0000AD6A C0EC04                 <1>     shr     ah, 4
0000AD6D D50A                   <1>     aad
0000AD6F 88C6                   <1>     mov     dh, al ; -> seconds
                                <1>
0000AD71 88E8                   <1>     mov     al, ch ; hours
                                <1>     ;shl    ax, 6
                                <1>     ; 29/07/2022
0000AD73 C1E006                 <1>     shl     eax, 6
0000AD76 08C8                   <1>     or      al, cl ; minutes
                                <1>     ;shl    ax, 5
                                <1>     ; 29/07/2022
0000AD78 C1E005                 <1>     shl     eax, 5
0000AD7B D0EE                   <1>     shr     dh, 1 ; 2 seconds
                                <1>     ; There is a bug in TRDOS v1 here !
                                <1>     ; it was 'or al, dl' !
0000AD7D 08F0                   <1>     or      al, dh ; seconds
                                <1>
                                <1>     ;pop    dx ; pop date
                                <1>     ; 29/07/2022
0000AD7F 5A                     <1>     pop     edx
                                <1>
0000AD80 C3                     <1>     retn
                                <1>
                                <1> save_directory_buffer:
                                <1>     ; 19/12/2025 (TRDOS 386 Kernel v2.0.10)
                                <1>     ; 30/07/2022
                                <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <1>     ; 15/10/2016
                                <1>     ; 23/03/2016
                                <1>     ; 26/02/2016
                                <1>     ; 22/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>     ; 01/08/2011
                                <1>     ; 14/03/2010
                                <1>     ; INPUT ->
                                <1>     ;       none
                                <1>     ; OUTPUT ->
                                <1>     ;  cf = 0 -> write OK...
                                <1>     ;  cf = 1 -> error code in AL (EAX)
                                <1>     ;  cf = 1 & AL = 0Dh => CH & CL = FS & FAT type
                                <1>     ;  EBX = Directory Buffer Address
                                <1>     ;
                                <1>     ;  (EAX, ECX, EDX will be modified)
                                <1>
0000AD81 BB00000800             <1>     mov     ebx, Directory_Buffer
0000AD86 803D[697E0100]02       <1>     cmp     byte [DirBuff_ValidData], 2
0000AD8D 7403                   <1>     je      short loc_save_dir_buffer
0000AD8F 31C0                   <1>     xor     eax, eax
0000AD91 C3                     <1>     retn
                                <1>
                                <1> loc_save_dir_buffer:
0000AD92 56                     <1>     push    esi
0000AD93 31DB                   <1>     xor     ebx, ebx
0000AD95 8A3D[677E0100]         <1>         mov     bh, [DirBuff_DRV]
0000AD9B 80EF41                 <1>     sub     bh, 'A'
0000AD9E BE00010900             <1>         mov     esi, Logical_DOSDisks
0000ADA3 01DE                   <1>     add     esi, ebx
                                <1>
                                <1> ; 19/12/2025
                                <1> %if 0
                                <1>         mov cx, [esi+LD_FATType]
                                <1>     ; CH = FS Type (A1h for FS)
                                <1>     ; CL = FAT Type (0 for FS)
                                <1>     or      cl, cl
                                <1>     jz      short loc_save_dir_buff_stc_retn
                                <1> %endif
                                <1>
                                <1> loc_save_dir_buffer_check_cluster_no:
0000ADA5 A1[6E7E0100]           <1>     mov     eax, [DirBuff_Cluster]
0000ADAA 28FF                   <1>     sub     bh, bh ; ebx = 0
0000ADAC 09C0                   <1>     or      eax, eax
0000ADAE 7537                   <1>     jnz     short loc_save_sub_dir_buffer
0000ADB0 8A25[687E0100]         <1>     mov     ah, [DirBuff_FATType]
```

```
                                <1>
                                <1> ; 19/12/2025
                                <1> %if 0
                                <1>     inc     bl ; bl = 1
                                <1>     cmp     ah, bl
                                <1>     jb      short loc_save_dir_buff_inv_data_retn
                                <1>     inc     bl ; bl = 2
                                <1>     cmp     bl, ah
                                <1>     jb      short loc_save_dir_buff_inv_data_retn
                                <1> %else
                                <1>     ; 19/12/2025
0000ADB6 80FC03                 <1>     cmp     ah, 3 ; FAT32 ?
0000ADB9 7314                   <1>     jnb     short loc_save_dir_buff_stc_retn ; yes
                                <1> %endif
                                <1>
                                <1> loc_save_root_dir_buffer:
0000ADBB 668B5E17               <1>     mov     bx, [esi+LD_BPB+RootDirEnts]
0000ADBF 6683C30F               <1>     add     bx, 15
                                <1>     ;shr    bx, 4 ; 16 dir entries per sector
                                <1>     ; 29/07/2022
0000ADC3 C1EB04                 <1>     shr     ebx, 4
                                <1>     ;or     bx, bx
0000ADC6 09DB                   <1>     or      ebx, ebx
0000ADC8 7405                   <1>     jz      short loc_save_dir_buff_stc_retn
                                <1>     ;mov    ecx, ebx
0000ADCA 8B4664                 <1>     mov     eax, [esi+LD_ROOTBegin] ; 26/02/2016
0000ADCD EB22                   <1>     jmp     short loc_write_directory_to_disk
                                <1>
                                <1> loc_save_dir_buff_stc_retn:
0000ADCF F9                     <1>     stc
                                <1> loc_save_dir_buff_inv_data_retn:
                                <1>     ; 15/10/2016 (0Dh -> 29)
0000ADD0 B01D                   <1>     mov     al, 29 ; Invalid data !
0000ADD2 C605[697E0100]00       <1>     mov     byte [DirBuff_ValidData], 0
0000ADD9 EB05                   <1>     jmp     short loc_save_dir_buff_retn
                                <1>
                                <1> loc_write_directory_to_disk_err:
                                <1>     ; 15/10/2016 (disk write error code, 1Dh -> 18)
0000ADDB B812000000             <1>     mov     eax, 18 ; Drive not ready or write error
                                <1>
                                <1> loc_save_dir_buff_retn:
0000ADE0 BB00000800             <1>     mov     ebx, Directory_Buffer
0000ADE5 5E                     <1>     pop     esi
0000ADE6 C3                     <1>     retn
                                <1>
                                <1> loc_save_sub_dir_buffer:
                                <1>     ; ebx = 0
                                <1>     ;sub    eax, 2
                                <1>     ; 30/07/2022
0000ADE7 48                     <1>     dec     eax
0000ADE8 48                     <1>     dec     eax
0000ADE9 8A5E13                 <1>     mov     bl, [esi+LD_BPB+SecPerClust]
0000ADEC F7E3                   <1>     mul     ebx
0000ADEE 034668                 <1>         add     eax, [esi+LD_DATABegin]
                                <1>     ;mov    ecx, ebx
                                <1>
                                <1> loc_write_directory_to_disk:
0000ADF1 89D9                   <1>     mov     ecx, ebx
0000ADF3 BB00000800             <1>     mov     ebx, Directory_Buffer
0000ADF8 E8B66B0000             <1>     call    disk_write
0000ADFD 72DC                   <1>     jc      short loc_write_directory_to_disk_err
                                <1>
                                <1> loc_save_dir_buff_validate_retn:
0000ADFF C605[697E0100]01       <1>     mov     byte [DirBuff_ValidData], 1
0000AE06 31C0                   <1>     xor     eax, eax
                                <1>     ; 26/02/2016
0000AE08 EBD6                   <1>     jmp     short loc_save_dir_buff_retn
                                <1>
                                <1> update_parent_dir_lmdt:
                                <1>     ; 19/12/2025
                                <1>     ; 16/07/2025 (TRDOS 386 Kernel v2.0.10) (BugFix)
                                <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <1>     ; 29/12/2017
                                <1>     ; 22/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>     ; 01/08/2011
                                <1>     ; 16/10/2010
                                <1>     ;
                                <1>     ; INPUT ->
                                <1>     ;       none
                                <1>     ; OUTPUT ->
                                <1>     ;       (last modification date & time of the parent dir
                                <1>     ;       will be changed/updated)
                                <1>     ;
                                <1>     ; (EAX, EBX, ECX, EDX, EDI will be changed)
                                <1>
0000AE0A 29C0                   <1>     sub     eax, eax
0000AE0C 8A25[40770100]         <1>     mov     ah, [Current_Dir_Level]
                                <1>
                                <1> ; 19/12/2025
                                <1> %if 0
                                <1>     mov     al, [Current_FATType]
                                <1>     cmp     al, 1
                                <1>     jb      short loc_UPDLMDT_proc_retn
                                <1> %endif
                                <1>
                                <1> loc_update_parent_dir_lm_date_time:
0000AE12 08E4                   <1>     or      ah, ah
0000AE14 7433                   <1>     jz      short loc_UPDLMDT_proc_retn
                                <1>
0000AE16 56                     <1>     push    esi ; *
0000AE17 8825[AB800100]         <1>     mov     [UPDLMDT_CDirLevel], ah
0000AE1D 8B15[3C770100]         <1>     mov     edx, [Current_Dir_FCluster]
0000AE23 8915[AC800100]         <1>     mov     [UPDLMDT_CDirFCluster], edx
                                <1>
0000AE29 FECC                   <1>     dec     ah
                                <1>     ;mov    ecx, 12
                                <1>         ; 29/07/2022
0000AE2B 29C9                   <1>     sub     ecx, ecx
0000AE2D B10C                   <1>     mov     cl, 12
                                <1>     ;
0000AE2F BE[A07E0100]           <1>     mov     esi, PATH_Array
                                <1>
0000AE34 8825[40770100]         <1>     mov     [Current_Dir_Level], ah
0000AE3A 08E4                   <1>     or      ah, ah
0000AE3C 750C                   <1>     jnz     short loc_update_parent_dir_lmdt_load_sub_dir_1
0000AE3E 803D[41770100]02       <1>     cmp     byte [Current_FATType], 2
0000AE45 7709                   <1>     ja      short loc_update_parent_dir_lmdt_load_sub_dir_2
                                <1>     ; 19/12/2025
                                <1>     ;sub    al, al
                                <1>     ; eax = 0
0000AE47 EB0A                   <1>     jmp     short loc_update_parent_dir_lmdt_load_sub_dir_3
                                <1>
                                <1> loc_UPDLMDT_proc_retn:
0000AE49 C3                     <1>     retn
```

```
2462                                <1>
2463                                <1> loc_update_parent_dir_lmdt_load_sub_dir_1:
2464 0000AE4A B010                  <1>     mov     al, 16
2465 0000AE4C F6E4                  <1>     mul     ah
2466 0000AE4E 01C6                  <1>     add     esi, eax
2467                                <1>
2468                                <1> loc_update_parent_dir_lmdt_load_sub_dir_2:
2469 0000AE50 8B460C                <1>     mov     eax, [esi+12] ; Parent Dir First Cluster
2470                                <1>
2471                                <1> loc_update_parent_dir_lmdt_load_sub_dir_3:
2472 0000AE53 A3[3C770100]          <1>     mov     [Current_Dir_FCluster], eax
2473                                <1>
2474 0000AE58 83C610                <1>     add     esi, 16
2475                                <1>     ;mov    di, Dir_File_Name
2476                                <1>     ; 16/07/2025 (BugFix)
2477 0000AE5B BF[C67F0100]          <1>     mov     edi, Dir_File_Name
2478 0000AE60 F3A4                  <1>     rep     movsb
2479                                <1>
2480 0000AE62 BE00010900            <1>     mov     esi, Logical_DOSDisks
2481 0000AE67 29DB                  <1>     sub     ebx, ebx
2482 0000AE69 8A3D[42770100]        <1>     mov     bh, [Current_Drv]
2483 0000AE6F 01DE                  <1>     add     esi, ebx
2484 0000AE71 E83BF8FFFF            <1>     call    reload_current_directory
2485 0000AE76 722F                  <1>     jc      short loc_update_parent_dir_lmdt_restore_cdirlevel
2486                                <1>
2487                                <1> loc_update_parent_dir_lmdt_locate_dir:
2488 0000AE78 BE[C67F0100]          <1>     mov     esi, Dir_File_Name
2489                                <1>     ;xor    cx, cx
2490                                <1>     ; 29/07/2022
2491 0000AE7D 31C9                  <1>     xor     ecx, ecx
2492 0000AE7F 66B81008              <1>     mov     ax, 0810h ; Only directories
2493 0000AE83 E870F7FFFF            <1>         call    locate_current_dir_file
2494                                <1>     ; EDI = DirBuff Directory Entry Address
2495 0000AE88 721D                  <1>     jc      short loc_update_parent_dir_lmdt_restore_cdirlevel
2496                                <1>
2497 0000AE8A E85DFEFFFF            <1>     call    convert_current_date_time
2498 0000AE8F 66895712              <1>     mov     [edi+18], dx ; Last Access Date
2499 0000AE93 66895718              <1>     mov     [edi+24], dx ; Last Write Date
2500 0000AE97 66894716              <1>     mov     [edi+22], ax ; Last Write Time
2501                                <1>
2502 0000AE9B C605[697E0100]02      <1>     mov     byte [DirBuff_ValidData], 2
2503 0000AEA2 E8DAFEFFFF            <1>     call    save_directory_buffer
2504                                <1>     ; 29/12/2017
2505                                <1>     ;jc     short loc_update_parent_dir_lmdt_restore_cdirlevel
2506                                <1>     ;xor    al, al
2507                                <1> loc_update_parent_dir_lmdt_restore_cdirlevel:
2508                                <1>     ;current directory level restoration
2509 0000AEA7 8A25[AB800100]        <1>     mov     ah, [UPDLMDT_CDirLevel]
2510 0000AEAD 8825[40770100]        <1>     mov     [Current_Dir_Level], ah
2511 0000AEB3 8B15[AC800100]        <1>         mov     edx, [UPDLMDT_CDirFCluster]
2512 0000AEB9 8915[3C770100]        <1>     mov     [Current_Dir_FCluster], edx
2513                                <1>
2514 0000AEBF 5E                    <1>     pop     esi ; *
2515 0000AEC0 C3                    <1>     retn
2516                                <1>
2517                                <1> delete_longname:
2518                                <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
2519                                <1>     ; 27/02/2016 (TRDOS 386 = TRDOS v2.0)
2520                                <1>     ; 01/08/2011 (DIR.ASM, 'proc_delete_longname')
2521                                <1>     ; 14/03/2010
2522                                <1>     ; INPUT ->
2523                                <1>     ;       EAX = Directory Entry (Index) Number (< 65536)
2524                                <1>     ; OUTPUT ->
2525                                <1>     ;       cf = 0 -> OK  (EAX = 0)
2526                                <1>     ;       cf = 1 -> error code in EAX (AL)
2527                                <1>     ;
2528                                <1>     ; (Modified registers: EAX, EDX, ECX, EBX, EDI)
2529                                <1>
2530 0000AEC1 66A3[14810100]        <1>     mov     [DLN_EntryNumber], ax
2531 0000AEC7 C605[16810100]40      <1>         mov     byte [DLN_40h], 40h
2532                                <1>
2533 0000AECE E857000000            <1>     call    locate_current_dir_entry
2534 0000AED3 7307                  <1>     jnc     short loc_dln_check_attributes
2535 0000AED5 C3                    <1>     retn
2536                                <1>
2537                                <1> loc_dln_longname_not_found:
2538                                <1>     ;mov    eax, 2
2539                                <1>     ; 29/07/2022
2540 0000AED6 29C0                  <1>     sub     eax, eax
2541 0000AED8 B002                  <1>     mov     al, 2
2542 0000AEDA F9                    <1>     stc
2543 0000AEDB C3                    <1>     retn
2544                                <1>
2545                                <1> loc_dln_check_attributes:
2546 0000AEDC B00F                  <1>     mov     al, 0Fh  ; long name
2547 0000AEDE 8A670B                <1>     mov     ah, [edi+0Bh] ; dir entry attributes
2548 0000AEE1 38C4                  <1>     cmp     ah, al
2549 0000AEE3 75F1                  <1>     jne     short loc_dln_longname_not_found
2550 0000AEE5 8A27                  <1>     mov     ah, [edi]
2551 0000AEE7 2A25[16810100]        <1>     sub     ah, [DLN_40h]
2552 0000AEED 76E7                  <1>     jna     short loc_dln_longname_not_found
2553 0000AEEF 80FC14                <1>     cmp     ah, 14h ; 84-64=20 -> 20*13=260 bytes
2554 0000AEF2 77E2                  <1>     ja      short loc_dln_longname_not_found
2555                                <1>
2556 0000AEF4 C607E5                <1>     mov     byte [edi], 0E5h  ; deleted sign
2557 0000AEF7 C605[697E0100]02      <1>     mov     byte [DirBuff_ValidData], 2 ; changed/write sign
2558 0000AEFE C605[16810100]00      <1>     mov     byte [DLN_40h], 0 ; 40h -> 0
2559                                <1>
2560                                <1> loc_dln_delete_next_ln_entry:
2561 0000AF05 80FC01                <1>     cmp     ah, 1
2562 0000AF08 7616                  <1>     jna     short loc_dln_longname_retn
2563                                <1> loc_dln_delete_next_ln_entry_0:
2564 0000AF0A 66FF05[14810100]      <1>     inc     word [DLN_EntryNumber]
2565 0000AF11 0FB705[14810100]      <1>     movzx   eax, word [DLN_EntryNumber]
2566 0000AF18 E80D000000            <1>     call    locate_current_dir_entry
2567 0000AF1D 73BD                  <1>     jnc     short loc_dln_check_attributes
2568                                <1>
2569                                <1> loc_dln_longname_stc_retn:
2570 0000AF1F C3                    <1>     retn
2571                                <1>
2572                                <1> loc_dln_longname_retn:
2573                                <1>     ;cmp    byte [DirBuff_ValidData], 2
2574                                <1>     ;jne    short loc_dln_longname_retn_xor_eax
2575 0000AF20 E85CFEFFFF            <1>     call    save_directory_buffer
2576 0000AF25 72F8                  <1>     jc      short loc_dln_longname_stc_retn
2577                                <1>
2578                                <1> loc_dln_longname_retn_xor_eax:
2579 0000AF27 31C0                  <1>     xor     eax, eax
2580 0000AF29 C3                    <1>     retn
2581                                <1>
2582                                <1> locate_current_dir_entry:
2583                                <1>     ; 19/12/2025 (TRDOS 386 v2.0.10)
2584                                <1>     ; 30/07/2022
2585                                <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
```

```
2586                            <1>     ; 16/10/2016
2587                            <1>     ; 15/10/2016
2588                            <1>     ; 23/03/2016
2589                            <1>     ; 27/02/2016 (TRDOS 386 = TRDOS v2.0)
2590                            <1>     ; 01/08/2011 (DIR.ASM, 'proc_locate_current_dir_entry')
2591                            <1>     ; 07/03/2010
2592                            <1>     ; INPUT ->
2593                            <1>     ;       EAX = Directory Entry (Index) Number (< 65536)
2594                            <1>     ; OUTPUT ->
2595                            <1>     ;       EDI = Directory Entry Address
2596                            <1>     ;       EAX = Cluster Number of Directory Buffer
2597                            <1>     ;       EBX = Directory Buffer Entry Offset
2598                            <1>     ;       ECX = DirBuff Valid Data identifier (CL)
2599                            <1>     ;       If CF = 0 and CL = 2 then
2600                            <1>     ;          directory buffer modified and
2601                            <1>     ;          must be written to disk.
2602                            <1>     ;       If CF = 0  and CL = 1 then
2603                            <1>     ;          dir buffer has been written to disk, already.
2604                            <1>     ;       CF = 1 -> Error code in EAX (AL)
2605                            <1>     ;
2606                            <1>     ; (Modified registers: EAX, EDX, ECX, EBX, EDI)
2607                            <1>
2608                            <1> loc_locate_current_dir_entry:
2609 0000AF2A 56                <1>     push    esi
2610 0000AF2B 89C1              <1>     mov     ecx, eax
2611                            <1>     ;mov    edx, 32
2612                            <1>     ; 29/07/2022
2613 0000AF2D 29D2              <1>     sub     edx, edx
2614 0000AF2F B220              <1>     mov     dl, 32
2615 0000AF31 F7E2              <1>     mul     edx
2616 0000AF33 A3[20810100]      <1>     mov     [LCDE_ByteOffset], eax
2617 0000AF38 31DB              <1>     xor     ebx, ebx
2618 0000AF3A 8A3D[42770100]    <1>     mov     bh, [Current_Drv]
2619 0000AF40 A0[677E0100]      <1>         mov     al, [DirBuff_DRV]
2620 0000AF45 2C41              <1>     sub     al, 'A'
2621 0000AF47 BE00010900        <1>         mov     esi, Logical_DOSDisks
2622 0000AF4C 01DE              <1>     add     esi, ebx
2623 0000AF4E 38C7              <1>     cmp     bh, al
2624                            <1>     ;jne    loc_lcde_reload_current_directory
2625                            <1>     ; 29/07/2022
2626 0000AF50 7405              <1>     je      short loc_lcde_cdl_check
2627 0000AF52 E986000000        <1>     jmp     loc_lcde_reload_current_directory
2628                            <1> loc_lcde_cdl_check:
2629                            <1>     ; 29/07/2022
2630 0000AF57 31C0              <1>     xor     eax, eax
2631 0000AF59 803D[40770100]00  <1>     cmp     byte [Current_Dir_Level], 0
2632 0000AF60 7728              <1>     ja      short loc_lcde_calc_dirbuff_cluster_offset
2633                            <1>     ; 27/02/2016
2634                            <1>     ; TRDOS v1 has bug here for FAT32 fs !
2635                            <1>     ; (Root Directory Entries for FAT32 = 0)
2636 0000AF62 807E0303          <1>     cmp     byte [esi+LD_FATType], 3  ; FAT32
2637 0000AF66 7322              <1>     jnb     short loc_lcde_calc_dirbuff_cluster_offset
2638                            <1>
2639                            <1> loc_lcde_cdl_check_FAT12_16:
2640                            <1>     ; 29/07/2022
2641                            <1>     ;xor    eax, eax
2642 0000AF68 668B4617          <1>     mov     ax, [esi+LD_BPB+RootDirEnts]
2643                            <1>     ;dec    ax
2644 0000AF6C 48                <1>     dec     eax
2645                            <1>     ;xor    dx, dx
2646                            <1>     ;cmp    ax, cx ; cx = Directory Entry (Index) Number
2647                            <1>     ; 29/07/2022
2648 0000AF6D 39C8              <1>     cmp     eax, ecx
2649 0000AF6F 720E              <1>     jb      short loc_lcde_stc_12h_retn
2650 0000AF71 66890D[18810100]  <1>     mov     [LCDE_EntryIndex], cx
2651 0000AF78 31C0              <1>     xor     eax, eax
2652 0000AF7A E985000000        <1>         jmp     loc_lcde_check_dir_buffer_cluster
2653                            <1>
2654                            <1> loc_lcde_stc_12h_retn:
2655 0000AF7F 5E                <1>     pop     esi
2656 0000AF80 89CB              <1>     mov     ebx, ecx
2657 0000AF82 89D1              <1>     mov     ecx, edx
2658                            <1>     ; 16/10/2016 (12h -> 12)
2659 0000AF84 B80C000000        <1>     mov     eax, 12 ; No more files
2660 0000AF89 C3                <1>     retn
2661                            <1>
2662                            <1> loc_lcde_calc_dirbuff_cluster_offset:
2663                            <1>     ;mov    bl, [esi+LD_BPB+SecPerClust]
2664                            <1>     ;xor    bh, bh
2665                            <1>     ;mov    ax, [esi+LD_BPB+BytesPerSec]
2666                            <1>     ;mul    bx
2667                            <1>     ;;or    dx, dx ; If bytes per cluster > 32KB it is invalid
2668                            <1>     ;; 29/07/2022
2669                            <1>     ;or     dl, dl
2670                            <1>     ;jnz    short loc_lcde_invalid_format
2671                            <1>     ; 29/07/2022
2672 0000AF8A 31DB              <1>     xor     ebx, ebx
2673 0000AF8C 8A5E13            <1>     mov     bl, [esi+LD_BPB+SecPerClust]
2674                            <1>     ;mov    ax, [esi+LD_BPB+BytesPerSec]
2675                            <1>     ; 19/12/2025
2676 0000AF8F 66B80002          <1>     mov     ax, 512
2677 0000AF93 F7E3              <1>     mul     ebx
2678 0000AF95 89C1              <1>     mov     ecx, eax
2679                            <1>     ;mov    cx, ax ; BYTES PER CLUSTER
2680 0000AF97 A1[20810100]      <1>     mov     eax, [LCDE_ByteOffset]
2681                            <1>     ;sub    edx, edx
2682 0000AF9C F7F1              <1>     div     ecx
2683 0000AF9E 3DFFFF0000        <1>     cmp     eax, 65535
2684 0000AFA3 7746              <1>     ja      short loc_lcde_invalid_format
2685                            <1>
2686                            <1>     ; cluster sequence number of directory (< 65536)
2687 0000AFA5 66A3[1A810100]    <1>     mov     [LCDE_ClusterSN], ax
2688                            <1>
2689                            <1>     ;mov    ax, dx ; byte offset in cluster (directory buffer)
2690                            <1>     ; 29/07/2022
2691 0000AFAB 89D0              <1>     mov     eax, edx
2692                            <1>     ;mov    bx, 32  ; 1 dir entry = 32 bytes
2693 0000AFAD B320              <1>         mov bl, 32
2694                            <1>     ;sub    dx, dx  ; 0
2695                            <1>     ;div    bx
2696 0000AFAF 29D2              <1>     sub     edx, edx
2697 0000AFB1 F7F3              <1>     div     ebx
2698 0000AFB3 66A3[18810100]    <1>     mov     [LCDE_EntryIndex], ax ; dir entry index/sequence number
2699                            <1>                                   ; (in directory buffer/cluster)
2700                            <1> loc_lcde_get_current_sub_dir_fcluster:
2701 0000AFB9 A1[3C770100]      <1>     mov     eax, [Current_Dir_FCluster]
2702                            <1>
2703                            <1> loc_lcde_get_next_cluster:
2704 0000AFBE 66833D[1A810100]00 <1>     cmp     word [LCDE_ClusterSN], 0
2705 0000AFC6 763C              <1>     jna     short loc_lcde_check_dir_buffer_cluster
2706 0000AFC8 A3[1C810100]      <1>     mov     [LCDE_Cluster], eax
2707 0000AFCD E8A70F0000        <1>     call    get_next_cluster
2708 0000AFD2 721E              <1>     jc      short loc_lcde_check_gnc_error
2709 0000AFD4 66FF0D[1A810100]  <1>     dec     word [LCDE_ClusterSN]
```

```
2710 0000AFDB EBE1                <1>     jmp     short loc_lcde_get_next_cluster
2711                              <1>
2712                              <1> loc_lcde_reload_current_directory:
2713 0000AFDD 51                  <1>     push    ecx
2714 0000AFDE E8CEF6FFFF          <1>     call    reload_current_directory
2715 0000AFE3 59                  <1>     pop     ecx
2716                              <1>     ;jnc    loc_lcde_cdl_check
2717                              <1>     ;pop    esi
2718                              <1>     ;retn
2719                              <1>     ; 09/08/2022
2720 0000AFE4 727E                <1>     jc      short loc_lcde_retn
2721 0000AFE6 E96CFFFFFF          <1>     jmp     loc_lcde_cdl_check
2722                              <1>
2723                              <1> loc_lcde_invalid_format:
2724                              <1>     ; 15/10/2016 (0Bh -> 28)
2725                              <1>     ;mov    eax, 28 ; Invalid Format !
2726                              <1>     ; 29/07/2022
2727 0000AFEB 29C0                <1>     sub     eax, eax
2728 0000AFED B01C                <1>     mov     al, 28
2729                              <1> loc_lcde_drive_not_ready_read_err:
2730 0000AFEF F9                  <1>     stc
2731 0000AFF0 5E                  <1>     pop     esi
2732 0000AFF1 C3                  <1>     retn
2733                              <1>
2734                              <1> loc_lcde_check_gnc_error:
2735 0000AFF2 09C0                <1>     or      eax, eax
2736 0000AFF4 75F9                <1>     jnz     short loc_lcde_drive_not_ready_read_err
2737 0000AFF6 66FF0D[1A810100]    <1>     dec     word [LCDE_ClusterSN]
2738 0000AFFD 75EC                <1>     jnz     short loc_lcde_invalid_format
2739 0000AFFF A1[1C810100]        <1>     mov     eax, [LCDE_Cluster]
2740                              <1>
2741                              <1> loc_lcde_check_dir_buffer_cluster:
2742 0000B004 3B05[6E7E0100]      <1>     cmp     eax, [DirBuff_Cluster]
2743 0000B00A 755A                <1>     jne     short loc_lcde_load_dir_cluster
2744 0000B00C 803D[697E0100]00    <1>     cmp     byte [DirBuff_ValidData], 0
2745 0000B013 7726                <1>     ja      short lcde_check_dir_buffer_cluster_next
2746 0000B015 803D[40770100]00    <1>     cmp     byte [Current_Dir_Level], 0
2747 0000B01C 775D                <1>     ja      short loc_lcde_load_dir_cluster_0
2748                              <1>     ; 27/02/2016
2749                              <1>     ; TRDOS v1 has bug here for FAT32 fs !
2750 0000B01E 807E0303            <1>     cmp     byte [esi+LD_FATType], 3  ; FAT32
2751 0000B022 7357                <1>     jnb     short loc_lcde_load_dir_cluster_0
2752                              <1>     ;
2753 0000B024 0FB74E17            <1>     movzx   ecx, word [esi+LD_BPB+RootDirEnts]
2754 0000B028 6683C10F            <1>     add     cx, 15 ; round up (16 entries per sector)
2755                              <1>     ;shr    cx, 4 ; 1 sector contains 16 dir entries
2756                              <1>     ; 29/07/2022
2757 0000B02C C1E904              <1>     shr     ecx, 4
2758 0000B02F 8B4664              <1>         mov     eax, [esi+LD_ROOTBegin]
2759 0000B032 EB52                <1>     jmp     short loc_lcde_load_dir_cluster_1
2760                              <1>
2761                              <1> loc_lcde_validate_dirBuff:
2762 0000B034 C605[697E0100]01    <1>     mov     byte [DirBuff_ValidData], 1
2763                              <1>
2764                              <1> lcde_check_dir_buffer_cluster_next:
2765 0000B03B 0FB71D[18810100]    <1>     movzx   ebx, word [LCDE_EntryIndex]
2766 0000B042 663B1D[6C7E0100]    <1>     cmp     bx, [DirBuff_LastEntry]
2767 0000B049 77A0                <1>     ja      short loc_lcde_invalid_format
2768                              <1>     ;mov    eax, 32
2769                              <1>     ; 29/07/2022
2770 0000B04B 31C0                <1>     xor     eax, eax
2771 0000B04D B020                <1>     mov     al, 32
2772 0000B04F F7E3                <1>     mul     ebx
2773                              <1>     ;or     edx, edx
2774                              <1>     ;jnz    short loc_lcde_invalid_format
2775                              <1>
2776 0000B051 BF00000800          <1>     mov     edi, Directory_Buffer
2777 0000B056 01C7                <1>     add     edi, eax ; add entry offset to buffer address
2778                              <1>
2779                              <1> loc_lcde_dir_buffer_last_check:
2780 0000B058 A1[6E7E0100]        <1>     mov     eax, [DirBuff_Cluster]
2781 0000B05D 0FB60D[697E0100]    <1>     movzx   ecx, byte [DirBuff_ValidData]
2782                              <1>
2783                              <1> loc_lcde_retn:
2784 0000B064 5E                  <1>     pop     esi
2785 0000B065 C3                  <1>     retn
2786                              <1>
2787                              <1> loc_lcde_load_dir_cluster:
2788                              <1>     ;cmp    byte [DirBuff_ValidData], 2
2789                              <1>     ;jne    short loc_lcde_load_dir_cluster_n2
2790 0000B066 50                  <1>     push    eax
2791 0000B067 E815FDFFFF          <1>     call    save_directory_buffer
2792 0000B06C 58                  <1>     pop     eax
2793 0000B06D 72F5                <1>     jc      short loc_lcde_retn
2794                              <1>
2795                              <1> loc_lcde_load_dir_cluster_n2:
2796 0000B06F C605[697E0100]00    <1>     mov     byte [DirBuff_ValidData], 0
2797 0000B076 A3[6E7E0100]        <1>     mov     [DirBuff_Cluster], eax
2798                              <1>
2799                              <1> loc_lcde_load_dir_cluster_0:
2800                              <1>     ;sub    eax, 2
2801                              <1>     ; 30/07/2022
2802 0000B07B 48                  <1>     dec     eax
2803 0000B07C 48                  <1>     dec     eax
2804 0000B07D 0FB64E13            <1>     movzx   ecx, byte [esi+LD_BPB+SecPerClust]
2805 0000B081 F7E1                <1>     mul     ecx
2806 0000B083 034668              <1>         add     eax, [esi+LD_DATABegin]
2807                              <1>
2808                              <1> loc_lcde_load_dir_cluster_1:
2809 0000B086 BB00000800          <1>     mov     ebx, Directory_Buffer
2810                              <1>     ; ecx = sector count
2811 0000B08B E832690000          <1>     call    disk_read
2812 0000B090 73A2                <1>     jnc     short loc_lcde_validate_dirBuff
2813                              <1>
2814                              <1>     ; 15/10/2016
2815                              <1>     ; (Disk read error instead of drv not ready err)
2816 0000B092 B811000000          <1>     mov     eax, 17 ; Drive not ready or read error !
2817 0000B097 EBCB                <1>     jmp     short loc_lcde_retn
2818                              <1>
2819                              <1> remove_file:
2820                              <1>     ; 19/12/2025 (TRDOS 386 Kernel v2.0.10)
2821                              <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
2822                              <1>     ; 15/10/2016
2823                              <1>     ; 28/02/2016 (TRDOS 386 = TRDOS v2.0)
2824                              <1>     ; 10/04/2011 (FILE.ASM, 'proc_delete_file')
2825                              <1>     ; 09/08/2010
2826                              <1>     ; INPUT ->
2827                              <1>     ;       EDI = Directory Buffer Entry Address
2828                              <1>     ;        CX = Directory Buffer Entry Counter/Index
2829                              <1>     ;        BL = Longname Entry Length
2830                              <1>     ;        BH = Logical DOS Drive Number
2831                              <1>
2832 0000B099 29C0                <1>     sub     eax, eax
2833 0000B09B 88FC                <1>     mov     ah, bh
```

```
2834 0000B09D BE00010900          <1>     mov     esi, Logical_DOSDisks
2835 0000B0A2 01C6                <1>     add     esi, eax
2836                              <1>
2837                              <1> ; 19/12/2025
2838                              <1> %if 0
2839                              <1>     cmp     byte [esi+LD_FATType], 1
2840                              <1>     jnb     short loc_del_fat_file
2841                              <1>
2842                              <1>     cmp     byte [esi+LD_FSType], 0A1h
2843                              <1>     je      short loc_del_fs_file
2844                              <1>
2845                              <1> loc_del_file_invalid_format:
2846                              <1>     xor     ah, ah
2847                              <1>     ; 15/10/2016 (0Bh -> 28)
2848                              <1>     mov     al, 28  ; Invalid Format
2849                              <1>     stc
2850                              <1>     retn
2851                              <1>
2852                              <1> loc_del_fs_file:
2853                              <1>     ;call  delete_fs_file
2854                              <1>     ;retn
2855                              <1>     ; 29/07/2022
2856                              <1>     jmp     delete_fs_file
2857                              <1> %endif
2858                              <1>
2859                              <1> loc_del_fat_file:
2860 0000B0A4 E808000000          <1>     call    delete_directory_entry
2861 0000B0A9 7205                <1>     jc      short loc_del_file_err_retn
2862                              <1>
2863                              <1> loc_delfile_unlink_cluster_chain:
2864 0000B0AB E85E160000          <1>     call    truncate_cluster_chain
2865                              <1>     ;jc     short loc_del_file_err_retn
2866                              <1>
2867                              <1> loc_delfile_return:
2868                              <1> loc_del_file_err_retn:
2869 0000B0B0 C3                  <1>     retn
2870                              <1>
2871                              <1> delete_directory_entry:
2872                              <1>     ; 15/10/2016
2873                              <1>     ; 28/02/2016 (TRDOS 386 = TRDOS v2.0)
2874                              <1>     ; 01/08/2011 (DIR.ASM, 'proc_delete_directory_entry')
2875                              <1>     ; 10/04/2011
2876                              <1>     ; INPUT ->
2877                              <1>     ;       ESI = Logical Dos Drive Descripton Table Address
2878                              <1>     ;       EDI = Directory Buffer Entry Address
2879                              <1>     ;        CX = Directory Buffer Entry Counter/Index
2880                              <1>     ;        BL = Longname Entry Length
2881                              <1>     ; OUTPUT ->
2882                              <1>     ;       ESI = Logical dos drive descripton table address
2883                              <1>     ;       EAX = First cluster to be truncated/unlinked
2884                              <1>     ;        CF = 1 -> Error code in EAX (AL)
2885                              <1>     ;        CF = 0 & BH <> 0 -> LMDT write error  (BH = 1)
2886                              <1>     ;        CF = 0 & BL <> 0 -> Long name delete error (BL = FFh)
2887                              <1>     ;
2888                              <1>     ;  (EDI, EBX, ECX register contents will be changed)
2889                              <1>
2890 0000B0B1 881D[AA800100]      <1>     mov     [DelFile_LNEL], bl
2891 0000B0B7 66890D[A8800100]    <1>     mov     [DelFile_EntryCounter], cx
2892                              <1>
2893 0000B0BE 668B4714            <1>     mov     ax, [edi+20] ; First Cluster High Word
2894 0000B0C2 C1E010              <1>     shl     eax, 16
2895 0000B0C5 668B471A            <1>     mov     ax, [edi+26] ; First Cluster Low Word
2896                              <1>
2897 0000B0C9 A3[A4800100]        <1>     mov     [DelFile_FCluster], eax
2898                              <1>
2899                              <1> loc_del_short_name:
2900 0000B0CE C607E5              <1>     mov     byte [edi], 0E5h  ; Deleted sign
2901                              <1>
2902 0000B0D1 C605[697E0100]02    <1>     mov     byte [DirBuff_ValidData], 2
2903 0000B0D8 E8A4FCFFFF          <1>     call    save_directory_buffer
2904 0000B0DD 723D                <1>     jc      short loc_delete_direntry_err_return
2905                              <1>
2906                              <1> loc_del_long_name:
2907 0000B0DF 0FB615[AA800100]    <1>     movzx   edx, byte [DelFile_LNEL]
2908 0000B0E6 08D2                <1>     or      dl, dl
2909 0000B0E8 7416                <1>     jz      short loc_del_dir_entry_update_parent_dir_lm_date
2910                              <1>
2911 0000B0EA 8835[AA800100]      <1>     mov     [DelFile_LNEL], dh ; 0
2912                              <1>
2913 0000B0F0 0FB705[A8800100]    <1>     movzx   eax, word [DelFile_EntryCounter]
2914 0000B0F7 29D0                <1>     sub     eax, edx
2915                              <1>     ;jnc    short loc_del_long_name_continue
2916 0000B0F9 7205                <1>     jc      short loc_del_dir_entry_update_parent_dir_lm_date
2917                              <1>
2918                              <1> ;loc_del_direntry_inv_data_return: ; 15/10/2016 (0Dh -> 29)
2919                              <1> ;   mov     eax, 29 ; 0Dh (TRDOS 8086) ; Invalid data
2920                              <1> ;   retn
2921                              <1>
2922                              <1> loc_del_long_name_continue:
2923                              <1>     ; AX = Directory Entry Number of the long name last entry
2924 0000B0FB E8C1FDFFFF          <1>     call    delete_longname
2925                              <1>     ;jc     short loc_delete_direntry_err_return
2926                              <1>
2927                              <1> loc_del_dir_entry_update_parent_dir_lm_date:
2928 0000B100 801D[AA800100]00    <1>     sbb     byte [DelFile_LNEL], 0 ; 0FFh if cf = 1
2929                              <1>
2930 0000B107 E8FEFCFFFF          <1>     call    update_parent_dir_lmdt
2931 0000B10C B700                <1>     mov     bh, 0
2932 0000B10E 80D700              <1>     adc     bh, 0
2933                              <1>
2934 0000B111 8A1D[AA800100]      <1>     mov     bl, byte [DelFile_LNEL]
2935                              <1>
2936                              <1> loc_delete_direntry_return:
2937 0000B117 A1[A4800100]        <1>     mov     eax, [DelFile_FCluster]
2938                              <1> loc_delete_direntry_err_return:
2939 0000B11C C3                  <1>     retn
2940                              <1>
2941                              <1> rename_directory_entry:
2942                              <1>     ; 19/12/2025 (TRDOS 386 Kernel v2.0.10)
2943                              <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
2944                              <1>     ; 13/11/2017
2945                              <1>     ; 15/10/2016
2946                              <1>     ; 06/03/2016 (TRDOS 386 = TRDOS v2.0)
2947                              <1>     ; 01/08/2011 (DIR.ASM, 'proc_rename_directory_entry')
2948                              <1>     ; 19/11/2010
2949                              <1>     ; INPUT -> (Current Directory)
2950                              <1>     ;       CX = Directory Entry Number
2951                              <1>     ;      EAX = First Cluster number of file or directory
2952                              <1>     ;      EBX = Longname Length (dir entry count) (< 256)
2953                              <1>     ;      ESI = New file (or directory) name (no path).
2954                              <1>     ;            (ASCIIZ string)
2955                              <1>     ; OUTPUT ->
2956                              <1>     ;      CF = 0 -> successfull
2957                              <1>     ;      CF = 1 -> error code in EAX (AL)
```

```
                                <1>     ;
                                <1>     ; (EAX, EBX, ECX, EDX, ESI, EDI will be changed)
                                <1>
                                <1> ; 19/12/2025
                                <1> %if 0
                                <1>     cmp     byte [Current_FATType], 0
                                <1>     ja      short loc_rename_directory_entry
                                <1>
                                <1>     ;call   rename_fs_file_or_directory
                                <1>     ;retn
                                <1>     ; 29/07/2022
                                <1>     jmp     rename_fs_file_or_directory
                                <1> %endif
                                <1>
                                <1> loc_rename_directory_entry:
0000B11D 881D[AA800100]         <1>     mov     [DelFile_LNEL], bl
0000B123 66890D[A8800100]       <1>     mov     [DelFile_EntryCounter], cx
0000B12A A3[A4800100]           <1>     mov     [DelFile_FCluster], eax
                                <1>
0000B12F 0FB7C1                 <1>     movzx   eax, cx
0000B132 E8F3FDFFFF             <1>     call    locate_current_dir_entry
0000B137 7307                   <1>     jnc     short loc_rename_direntry_check_fcluster
                                <1>
                                <1> loc_rename_direntry_pop_retn:
0000B139 C3                     <1>     retn
                                <1>
                                <1> loc_rename_direntry_pop_invd_retn:
                                <1>     ; 29/07/2022
                                <1>     ;stc
                                <1> loc_rename_direntry_invd_retn:
                                <1>     ; 15/10/2016 (0Dh -> 29)
                                <1>     ;mov    eax, 29 ; Invalid data
                                <1>     ; 29/07/2022
0000B13A 29C0                   <1>     sub     eax, eax
0000B13C B01D                   <1>     mov     al, 29
0000B13E F9                     <1>     stc
                                <1> loc_rename_retn:
0000B13F C3                     <1>     retn
                                <1>
                                <1> loc_rename_direntry_check_fcluster:
0000B140 668B5714               <1>     mov     dx, [edi+20] ; First Cluster HW
0000B144 C1E210                 <1>     shl     edx, 16 ; 13/11/2017
0000B147 668B571A               <1>     mov     dx, [edi+26] ; First Cluster LW
0000B14B 3B15[A4800100]         <1>     cmp     edx, [DelFile_FCluster]
0000B151 75E7                   <1>     jne     short loc_rename_direntry_pop_invd_retn
                                <1>     ; ESI = New file (or directory) name. (ASCIIZ string)
                                <1>     ; 06/03/2016
                                <1>     ; TRDOS v2 - NOTE: 'convert_file_name' procedure
                                <1>     ; has been modified for eliminating following situation.
                                <1>     ;
                                <1>     ; TRDOS v1 - NOTE: If file/dir name is more than 11 bytes
                                <1>     ; without a dot, attributes (edi+11) byte will be overwritten !
                                <1>     ; (Dot file name input must be proper for 11 byte dir entry
                                <1>     ;  type file name output.)
0000B153 E84CF7FFFF             <1>     call    convert_file_name
                                <1>
0000B158 C605[697E0100]02       <1>         mov     byte [DirBuff_ValidData], 2
0000B15F E81DFCFFFF             <1>     call    save_directory_buffer
0000B164 72D9                   <1>     jc      short loc_rename_retn
                                <1>
                                <1> loc_rename_direntry_del_ln:
0000B166 0FB615[AA800100]       <1>     movzx   edx, byte [DelFile_LNEL]
0000B16D 08D2                   <1>     or      dl, dl
0000B16F 7410                   <1>     jz      short loc_rename_direntry_update_parent_dir_lm_date
                                <1>
0000B171 0FB705[A8800100]       <1>     movzx   eax, word [DelFile_EntryCounter]
0000B178 29D0                   <1>     sub     eax, edx
0000B17A 72BE                   <1>     jc      short loc_rename_direntry_invd_retn
                                <1>
                                <1> loc_rename_direntry_del_ln_continue:
                                <1>     ; EAX = Directory Entry Number of the long name last entry
0000B17C E840FDFFFF             <1>     call    delete_longname
                                <1>
                                <1> loc_rename_direntry_update_parent_dir_lm_date:
0000B181 E884FCFFFF             <1>     call    update_parent_dir_lmdt
0000B186 31C0                   <1>     xor     eax, eax
0000B188 C3                     <1>     retn
                                <1>
                                <1> move_source_file_to_destination_file:
                                <1>     ; 19/12/2025
                                <1>     ; 11/08/2025 (TRDOS 386 Kernel v2.0.10)
                                <1>     ; 07/08/2022
                                <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <1>     ; 15/10/2016
                                <1>     ; 11/03/2016
                                <1>     ; 10/03/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>     ; 01/08/2011 (FILE.ASM)
                                <1>     ; 04/08/2010
                                <1>     ;
                                <1>     ;   Phase 1 -> Check destination file,
                                <1>     ;              'not found' is required
                                <1>     ;   Phase 2 -> Check source file
                                <1>     ;              'found' and proper attributes is required
                                <1>     ;   Phase 3 -> Make destination directory entry,
                                <1>     ;             add new dir cluster or section if it is required
                                <1>     ;   Phase 4 -> Delete source directory entry.
                                <1>     ;       cf = 1 causes to return before the phase 4.
                                <1>     ;    (source file protection against any possible errors)
                                <1>     ;
                                <1>     ; 08/05/2011 major modification
                                <1>     ;             -> destination file deleting is removed
                                <1>     ;             for msdos move/rename compatibility.
                                <1>     ;             (Access denied error will return if
                                <1>     ;             the destination file is found...)
                                <1>     ; INPUT ->
                                <1>     ;       ESI = Source File Pathname (Asciiz)
                                <1>     ;       EDI = Destination File Pathname (Asciiz)
                                <1>     ;       AL = 0 --> Interrupt (System call)
                                <1>     ;       AL > 0 --> Command Interpreter (Question)
                                <1>     ;       AL = 1 --> Question Phase
                                <1>     ;       AL = 2 --> Progress Phase
                                <1>     ; OUTPUT ->
                                <1>     ;       cf = 0 -> OK
                                <1>     ;       EAX = Destination directory first cluster
                                <1>     ;       ESI = Logical DOS drive description table
                                <1>     ;       EBX = Destination file structure offset
                                <1>     ;       CX = 0 (CX > 0 --> calculate free space error)
                                <1>     ;       cf = 1 -> Error code in EAX (AL)
                                <1>     ;
                                <1>     ;  (EDX, ECX, EBX, ESI, EDI will be changed)
                                <1>
0000B189 3C02                   <1>     cmp     al, 2
                                <1>     ;je     msftdf_df2_check_directory
                                <1>     ; 29/07/2022
```

```
3082 0000B18B 7505              <1>     jne     short msftdf
3083 0000B18D E96D010000        <1>     jmp     msftdf_df2_check_directory
3084                            <1> msftdf:
3085 0000B192 A2[2E820100]      <1>     mov     [move_cmd_phase], al
3086                            <1>
3087                            <1> msftdf_parse_sf_path:
3088                            <1>     ; ESI = ASCIIZ pathname (Source)
3089 0000B197 57                <1>     push    edi
3090 0000B198 BF[2C810100]      <1>     mov     edi, SourceFile_Drv
3091 0000B19D E8CCF7FFFF        <1>     call    parse_path_name
3092 0000B1A2 5E                <1>     pop     esi
3093 0000B1A3 7211              <1>     jc      short msftdf_psf_retn
3094                            <1>
3095                            <1> msftdf_parse_df_path:
3096                            <1>     ; ESI = ASCIIZ pathname       (Destination)
3097 0000B1A5 BF[AC810100]      <1>     mov     edi, DestinationFile_Drv
3098 0000B1AA E8BFF7FFFF        <1>     call    parse_path_name
3099 0000B1AF 7306              <1>     jnc     short msftdf_check_sf_drv
3100                            <1>
3101 0000B1B1 3C01              <1>     cmp     al, 1 ; File or directory name is not existing
3102 0000B1B3 7602              <1>     jna     short msftdf_check_sf_drv
3103                            <1>
3104                            <1> msftdf_stc_retn:
3105 0000B1B5 F9                <1>     stc
3106                            <1> msftdf_psf_retn:
3107 0000B1B6 C3                <1>     retn
3108                            <1>
3109                            <1> msftdf_check_sf_drv:
3110 0000B1B7 A0[2C810100]      <1>     mov     al, [SourceFile_Drv]
3111                            <1>
3112                            <1> msftdf_check_df_drv:
3113 0000B1BC 8A15[AC810100]    <1>     mov     dl, [DestinationFile_Drv]
3114                            <1>
3115                            <1> msftdf_compare_sf_df_drv:
3116                            <1>     ; 11/08/2025
3117                            <1>     ;sub    ebx, ebx
3118                            <1>     ;mov    bh, [Current_Drv]
3119 0000B1C2 38C2              <1>     cmp     dl, al
3120 0000B1C4 7408              <1>     je      short msftdf_check_sf_df_drv_ok
3121                            <1>
3122                            <1> msftdf_not_same_drv:
3123                            <1>         ; DL = source file's drive number
3124 0000B1C6 88C6              <1>     mov     dh, al ; destination file's drive number
3125                            <1>     ; 15/10/2016 (11h -> 21)
3126                            <1>     ;mov    eax, 21 ; Not the same drive
3127                            <1>     ; 29/07/2022
3128 0000B1C8 29C0              <1>     sub     eax, eax
3129 0000B1CA B015              <1>     mov     al, 21
3130 0000B1CC F9                <1>     stc
3131 0000B1CD C3                <1>     retn
3132                            <1>
3133                            <1> msftdf_check_sf_df_drv_ok:
3134 0000B1CE 8815[2F820100]    <1>     mov     [msftdf_sf_df_drv], dl
3135                            <1>
3136 0000B1D4 29C0              <1>         sub eax, eax
3137 0000B1D6 88D4              <1>     mov     ah, dl
3138 0000B1D8 0500010900        <1>     add     eax, Logical_DOSDisks
3139 0000B1DD A3[30820100]      <1>     mov     [msftdf_drv_offset], eax
3140                            <1>
3141                            <1>     ;cmp    dl, bh ; byte [Current_Drv]
3142                            <1>     ; 11/08/2025
3143 0000B1E2 3A15[42770100]    <1>     cmp     dl, [Current_Drv]
3144 0000B1E8 7407              <1>     je      short msftdf_df_check_directory
3145                            <1>
3146                            <1> msftdf_change_drv:
3147 0000B1EA E8F1C4FFFF        <1>     call    change_current_drive
3148 0000B1EF 726C              <1>     jc      short msftdf_df_error_retn
3149                            <1>
3150                            <1> msftdf_check_destination_file:
3151                            <1> msftdf_df_check_directory:
3152 0000B1F1 BE[AD810100]      <1>     mov     esi, DestinationFile_Directory
3153 0000B1F6 803E20            <1>     cmp     byte [esi], 20h
3154 0000B1F9 760F              <1>     jna     short msftdf_df_find_1
3155                            <1>
3156                            <1> msftdf_df_change_directory:
3157 0000B1FB FE05[5D2E0100]    <1>     inc     byte [Restore_CDIR]
3158 0000B201 30E4              <1>     xor     ah, ah ; CD_COMMAND sign -> 0
3159 0000B203 E8D3F1FFFF        <1>     call    change_current_directory
3160 0000B208 7253              <1>     jc      short msftdf_df_error_retn
3161                            <1>
3162                            <1> ;msftdf_df_change_prompt_dir_string:
3163                            <1> ;   call    change_prompt_dir_string
3164                            <1>
3165                            <1> msftdf_df_find_1:
3166 0000B20A BE[EE810100]      <1>         mov     esi, DestinationFile_Name
3167 0000B20F 803E20            <1>     cmp     byte [esi], 20h
3168 0000B212 7631              <1>     jna     short msftdf_df_copy_sf_name
3169                            <1>
3170                            <1> msftdf_df_find_2:
3171                            <1>     ;xor    ax, ax ; DestinationFile_AttributesMask -> any/zero
3172                            <1>     ; 11/08/2025
3173 0000B214 31C0              <1>     xor     eax, eax
3174 0000B216 E84FD7FFFF        <1>     call    find_first_file
3175                            <1>     ;jnc    msftdf_permission_denied_retn
3176                            <1>     ; 29/07/2022
3177 0000B21B 7205              <1>     jc      short msftdf_df_check_error_code
3178 0000B21D E98B000000        <1>     jmp     msftdf_permission_denied_retn
3179                            <1>
3180                            <1> msftdf_df_check_error_code:
3181                            <1>     ;cmp    eax, 2 ; File not found error
3182 0000B222 3C02              <1>     cmp     al, 2
3183 0000B224 7536              <1>     jne     short msftdf_df_stc_retn
3184                            <1>
3185                            <1> msftdf_df_check_fname:
3186                            <1>     ; 15/10/2016
3187 0000B226 BE[EE810100]      <1>     mov     esi, DestinationFile_Name ; *
3188 0000B22B E899DAFFFF        <1>     call    check_filename
3189 0000B230 7307              <1>     jnc     short msftdf_convert_df_direntry_name
3190                            <1>     ; invalid file name chars !
3191 0000B232 B81A000000        <1>     mov     eax, ERR_INV_FILE_NAME  ; 26
3192 0000B237 EB23              <1>     jmp     short msftdf_df_stc_retn
3193                            <1>
3194                            <1> msftdf_convert_df_direntry_name:
3195                            <1>     ;mov    esi, DestinationFile_Name ; *
3196 0000B239 BF[FE810100]      <1>     mov     edi, DestinationFile_DirEntry
3197 0000B23E E861F6FFFF        <1>     call    convert_file_name
3198 0000B243 EB19              <1>     jmp     short msftdf_restore_current_dir_1
3199                            <1>
3200                            <1> msftdf_df_copy_sf_name:
3201 0000B245 89F7              <1>     mov     edi, esi
3202 0000B247 57                <1>     push    edi
3203 0000B248 BE[6E810100]      <1>         mov     esi, SourceFile_Name
3204                            <1>     ;mov    ecx, 12
3205                            <1>     ; 29/07/2022
```

```
                        <1>     sub     ecx, ecx
                        <1>     mov     cl, 12
                        <1> msftdf_df_copy_sf_name_loop:
0000B251 AC             <1>     lodsb
0000B252 AA             <1>         stosb
0000B253 08C0           <1>     or      al, al
0000B255 7402           <1>     jz      short msftdf_df_copy_sf_name_ok
0000B257 E2F8           <1>         loop    msftdf_df_copy_sf_name_loop
                        <1> msftdf_df_copy_sf_name_ok:
0000B259 5E             <1>     pop     esi
0000B25A EBB8           <1>     jmp     short msftdf_df_find_2
                        <1>
                        <1> msftdf_df_stc_retn:
0000B25C F9             <1>     stc
                        <1> msftdf_restore_cdir_failed:
                        <1> msftdf_df_error_retn:
0000B25D C3             <1>     retn
                        <1>
                        <1> msftdf_restore_current_dir_1:
0000B25E 803D[5D2E0100]00 <1>     cmp     byte [Restore_CDIR], 0
0000B265 760D           <1>     jna     short msftdf_sf_check_directory
0000B267 8B35[30820100] <1>     mov     esi, [msftdf_drv_offset]
0000B26D E81FC5FFFF     <1>     call    restore_current_directory
0000B272 72E9           <1>     jc      short msftdf_restore_cdir_failed
                        <1>
                        <1> msftdf_sf_check_directory:
0000B274 BE[2D810100]   <1>     mov     esi, SourceFile_Directory
0000B279 803E20         <1>     cmp     byte [esi], 20h
0000B27C 760F           <1>     jna     short msftdf_sf_find
                        <1> msftdf_sf_change_directory:
0000B27E FE05[5D2E0100] <1>     inc     byte [Restore_CDIR]
0000B284 30E4           <1>     xor     ah, ah ; CD_COMMAND sign -> 0
0000B286 E850F1FFFF     <1>     call    change_current_directory
0000B28B 7225           <1>     jc      short msftdf_return
                        <1>
                        <1> ;msftdf_sf_change_prompt_dir_string:
                        <1> ;   call    change_prompt_dir_string
                        <1>
0000B28D BE[6E810100]   <1> msftdf_sf_find:
                        <1>         mov     esi, SourceFile_Name  ; Offset 66
0000B292 66B80018       <1>     mov     ax, 1800h ; Only files
0000B296 E8CFD6FFFF     <1>     call    find_first_file
0000B29B 7215           <1>     jc      short msftdf_return
                        <1>
                        <1> msftdf_sf_ambgfn_check:
0000B29D 6609D2         <1>     or      dx, dx ; Ambiguous filename chars used sign (DX>0)
0000B2A0 7406           <1>     jz      short msftdf_sf_found
                        <1>
                        <1> msftdf_ambiguous_file_name_error:
                        <1>     ;mov    eax, 2 ; File not found error
                        <1>     ; 29/07/2022
0000B2A2 29C0           <1>     sub     eax, eax
0000B2A4 B002           <1>     mov     al, 2
0000B2A6 F9             <1>     stc
0000B2A7 C3             <1>     retn
                        <1>
                        <1> msftdf_sf_found:
0000B2A8 80E31F         <1>     and     bl, 1Fh ; Attributes, D-V-S-H-R
0000B2AB 7415           <1>     jz      short msftdf_save_sf_structure
                        <1>
                        <1> msftdf_permission_denied_retn:
                        <1>     ;mov    eax, 05h ; Access (Permission) denied !
                        <1>     ; 29/07/2022
0000B2AD 29C0           <1>     sub     eax, eax
0000B2AF B005           <1>     mov     al, 5
0000B2B1 F9             <1>     stc
                        <1> msftdf_rest_cdir_err_retn:
                        <1> msftdf_return:
0000B2B2 C3             <1>     retn
                        <1>
                        <1> msftdf_phase_1_return:
0000B2B3 31C0           <1>     xor     eax, eax
0000B2B5 A2[2E820100]   <1>     mov     [move_cmd_phase], al ; 0
0000B2BA FEC0           <1>     inc     al ; mov al, 1
0000B2BC BB[FFB20000]   <1>     mov     ebx, msftdf_df2_check_directory
                        <1>     ;mov    edx, 0FFFFFFFFh
0000B2C1 C3             <1>     retn
                        <1>
                        <1> msftdf_save_sf_structure:
0000B2C2 BE[34800100]   <1>     mov     esi, FindFile_DirEntry
0000B2C7 BF[7E810100]   <1>     mov     edi, SourceFile_DirEntry
                        <1>     ;mov    ecx, 8
                        <1>     ; 29/07/2022
0000B2CC 29C9           <1>     sub     ecx, ecx
0000B2CE B108           <1>     mov     cl, 8
0000B2D0 F3A5           <1>     rep     movsd
                        <1>
                        <1> msftdf_df_copy_sf_parameters:
                        <1>     ;mov    esi, 11
                        <1>     ;mov    edi, esi
                        <1>     ;add    esi, SourceFile_DirEntry
                        <1>     ;add    edi, DestinationFile_DirEntry
                        <1>     ; 11/08/2025
0000B2D2 BE[89810100]   <1>     mov     esi, SourceFile_DirEntry+11
0000B2D7 BF[09820100]   <1>     mov     edi, DestinationFile_DirEntry+11
                        <1>     ;mov    ecx, 21
0000B2DC B115           <1>     mov     cl, 21
0000B2DE F3A4           <1>     rep     movsb
                        <1>
                        <1> msftdf_restore_current_dir_2:
0000B2E0 803D[5D2E0100]00 <1>     cmp     byte [Restore_CDIR], 0
0000B2E7 760D           <1>     jna     short msftdf_df2_check_move_cmd_phase
0000B2E9 8B35[30820100] <1>     mov     esi, [msftdf_drv_offset]
0000B2EF E89DC4FFFF     <1>     call    restore_current_directory
0000B2F4 72BC           <1>     jc      short msftdf_rest_cdir_err_retn
                        <1>
                        <1> msftdf_df2_check_move_cmd_phase:
0000B2F6 803D[2E820100]01 <1>     cmp     byte [move_cmd_phase], 1
0000B2FD 74B4           <1>     je      short msftdf_phase_1_return
                        <1>
                        <1> msftdf_df2_check_directory:
0000B2FF BE[AD810100]   <1>     mov     esi, DestinationFile_Directory
0000B304 803E20         <1>     cmp     byte [esi], 20h
0000B307 760F           <1>     jna     short msftdf_make_dfde_locate_ffe_on_directory
                        <1> msftdf_df2_change_directory:
0000B309 FE05[5D2E0100] <1>     inc     byte [Restore_CDIR]
0000B30F 30E4           <1>     xor     ah, ah ; CD_COMMAND sign -> 0
0000B311 E8C5F0FFFF     <1>     call    change_current_directory
0000B316 729A           <1>     jc      short msftdf_return
                        <1>
                        <1> ;msftdf_df2_change_prompt_dir_string:
                        <1> ;   call    change_prompt_dir_string
                        <1>
                        <1> msftdf_make_dfde_locate_ffe_on_directory:
```

```
                                <1>     ; Current directory fcluster <> Directory buffer cluster
                                <1>     ; Current directory will be reloaded by
                                <1>     ; 'locate_current_dir_file' procedure
                                <1>     ;
                                <1>     ;xor    ax, ax
0000B318 31C0                   <1>     xor     eax, eax
0000B31A 89C1                   <1>     mov     ecx, eax
0000B31C 6649                   <1>     dec     cx ; FFFFh
                                <1>             ; CX = FFFFh -> find first deleted or free entry
                                <1>             ; ESI would be ASCIIZ filename address if the call
                                <1>             ; would not be for first free or deleted dir entry
0000B31E E8D5F2FFFF             <1>     call    locate_current_dir_file
                                <1>     ; 07/08/2022
0000B323 7328                   <1>     jnc     short msftdf_make_dfde_set_ff_dir_entry
                                <1>
                                <1>     ;cmp    eax, 2
0000B325 3C02                   <1>         cmp al, 2
0000B327 7520                   <1>     jne     short msftdf_error_retn
                                <1>
                                <1> msftdf_add_new_dir_entry_check_fs:
0000B329 8B35[30820100]         <1>     mov     esi, [msftdf_drv_offset]
0000B32F A1[6E7E0100]           <1>     mov     eax, [DirBuff_Cluster]
                                <1>
                                <1> ; 19/12/2025
                                <1> %if 0
                                <1>     cmp     byte [esi+LD_FATType], 0
                                <1>     ja      short msftdf_add_new_subdir_cluster
                                <1>
                                <1> msftdf_add_new_fs_subdir_section:
                                <1>     ;CL=0, CH=E5h --> deleted entry, CH=0 --> free entry
                                <1>         ;xor       cx, cx
                                <1>     xor     ch, ch ; cx = 0 --> add a new subdir section
                                <1>     call    add_new_fs_section
                                <1>         jc  short msftdf_dsfde_error_retn
                                <1>     ;mov    [createfile_LastDirCluster], eax
                                <1>
                                <1>     call    load_FS_sub_directory
                                <1>     ;mov    ebx, Directory_Buffer
                                <1>     jnc     short msftdf_add_new_fs_subdir_section_ok
                                <1>     retn
                                <1> %endif
                                <1>
                                <1> msftdf_add_new_subdir_cluster:
0000B334 E8C3140000             <1>     call    add_new_cluster
0000B339 720F                   <1>     jc      short msftdf_dsfde_error_retn
                                <1>
                                <1>     ;mov    [createfile_LastDirCluster], eax
                                <1>
0000B33B E8F50D0000             <1>     call    load_FAT_sub_directory
0000B340 7309                   <1>     jnc     short msftdf_add_new_subdir_cluster_ok
                                <1>     ; EBX = Directory buffer address
                                <1>
                                <1> msftdf_ansdc_update_parent_dir_lmdt:
                                <1> msftdf_make_dfde_err_upd_pdir_lmdt:
0000B342 50                     <1>     push    eax
0000B343 E8C2FAFFFF             <1>     call    update_parent_dir_lmdt
0000B348 58                     <1>     pop     eax
                                <1>
                                <1> msftdf_error_retn:
0000B349 F9                     <1>     stc
                                <1> msftdf_dsfde_restore_cdir_failed:
                                <1> msftdf_dsfde_error_retn:
0000B34A C3                     <1>     retn
                                <1>
                                <1> msftdf_add_new_fs_subdir_section_ok:
                                <1> msftdf_add_new_subdir_cluster_ok:
0000B34B 89DF                   <1>     mov     edi, ebx ; Directory buffer address
                                <1>
                                <1> msftdf_make_dfde_set_ff_dir_entry:
0000B34D 8B15[3C770100]         <1>     mov     edx, [Current_Dir_FCluster]
0000B353 8915[94820100]         <1>     mov     [createfile_FFCluster], edx
                                <1>     ; EDI = Directory entry offset
0000B359 BE[FE810100]           <1>     mov     esi, DestinationFile_DirEntry
                                <1>     ;mov    ecx, 8
                                <1>     ; 29/07/2022
0000B35E 29C9                   <1>     sub     ecx, ecx
0000B360 B108                   <1>     mov     cl, 8
0000B362 F3A5                   <1>     rep     movsd
                                <1>
0000B364 C605[697E0100]02       <1>     mov     byte [DirBuff_ValidData], 2
0000B36B E811FAFFFF             <1>     call    save_directory_buffer
0000B370 72D0                   <1>     jc      short msftdf_make_dfde_err_upd_pdir_lmdt
                                <1>
                                <1> msftdf_make_dfde_update_pdir_lmdt:
0000B372 E893FAFFFF             <1>     call    update_parent_dir_lmdt
                                <1>
                                <1> msftdf_dsfde_restore_current_dir_1:
0000B377 803D[5D2E0100]00       <1>     cmp     byte [Restore_CDIR], 0
0000B37E 760D                   <1>     jna     short msftdf_dsfde_check_directory
0000B380 8B35[30820100]         <1>     mov     esi, [msftdf_drv_offset]
0000B386 E806C4FFFF             <1>     call    restore_current_directory
0000B38B 72BD                   <1>     jc      short msftdf_dsfde_restore_cdir_failed
                                <1>
                                <1> msftdf_dsfde_check_directory:
0000B38D BE[2D810100]           <1>     mov     esi, SourceFile_Directory
0000B392 803E20                 <1>     cmp     byte [esi], 20h
0000B395 760F                   <1>     jna     short msftdf_dsfde_find_file
                                <1>
                                <1> msftdf_dsfde_change_directory:
0000B397 FE05[5D2E0100]         <1>     inc     byte [Restore_CDIR]
0000B39D 28E4                   <1>     sub     ah, ah ; CD_COMMAND sign -> 0
0000B39F E837F0FFFF             <1>     call    change_current_directory
0000B3A4 72A4                   <1>     jc      short msftdf_dsfde_error_retn
                                <1>
                                <1> ;msftdf_dsfde_sf_change_prompt_dir_string:
                                <1> ;   call    change_prompt_dir_string
                                <1>
                                <1> msftdf_dsfde_find_file:
0000B3A6 BE[6E810100]           <1>     mov     esi, SourceFile_Name  ; Offset 66
0000B3AB 668B460E               <1>     mov     ax, [esi+14] ; 80 -> SourceFile_AttributesMask
0000B3AF E8B6D5FFFF             <1>     call    find_first_file
0000B3B4 7294                   <1>     jc      short msftdf_dsfde_error_retn
                                <1>
                                <1> msftdf_dsfde_delete_direntry:
0000B3B6 8B35[30820100]         <1>     mov     esi, [msftdf_drv_offset]
                                <1>
                                <1> ; 19/12/2025
                                <1> %if 0
                                <1>     cmp     byte [esi+LD_FATType], 0
                                <1>     ja      short msftdf_delete_FAT_direntry
                                <1>
                                <1>     xor     bl, bl
                                <1>     ; BL = 0 -> File
                                <1>     ; EDI -> Directory buffer entry offset/address
```

```
                                <1>     call    delete_fs_directory_entry
                                <1>     jnc     short msftdf_dsfde_restore_current_dir_2
                                <1>     retn
                                <1> %endif
                                <1>
                                <1> msftdf_delete_FAT_direntry:
0000B3BC 8A1D[31800100]         <1>     mov     bl, [FindFile_LongNameEntryLength]
0000B3C2 668B0D[5C800100]       <1>     mov     cx, [FindFile_DirEntryNumber]
                                <1>     ; ESI = Logical DOS drive description table address
                                <1>     ; EDI = Directory buffer entry offset/address
0000B3C9 E8E3FCFFFF             <1>     call    delete_directory_entry
0000B3CE 721C                   <1>     jc      short msftdf_retn
                                <1>
                                <1> msftdf_dsfde_restore_current_dir_2:
0000B3D0 803D[5D2E0100]00       <1>     cmp     byte [Restore_CDIR], 0
0000B3D7 7607                   <1>     jna     short msftdf_new_dir_fcluster_retn
                                <1>     ;mov    esi, [msftdf_drv_offset]
0000B3D9 E8B3C3FFFF             <1>     call    restore_current_directory
0000B3DE 720C                   <1>     jc      short msftdf_retn
                                <1>
                                <1> msftdf_new_dir_fcluster_retn:
0000B3E0 31C9                   <1>     xor     ecx, ecx
0000B3E2 A1[94820100]           <1>     mov     eax, [createfile_FFCluster]
0000B3E7 BB[AC810100]           <1>     mov     ebx, DestinationFile_Drv
                                <1>
                                <1> msftdf_retn:
0000B3EC C3                     <1>     retn
                                <1>
                                <1> copy_source_file_to_destination_file:
                                <1>     ; 19/12/2025 (TRDOS 386 Kernel v2.0.10)
                                <1>     ; 31/08/2024
                                <1>     ; 30/08/2024
                                <1>     ; 29/08/2024
                                <1>     ; 26/08/2024
                                <1>     ; 25/08/2024 (TRDOS 386 v2.0.9)
                                <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <1>     ; 17/10/2016
                                <1>     ; 16/10/2016
                                <1>     ; 15/10/2016
                                <1>     ; 30/03/2016, 31/03/2016
                                <1>     ; 24/03/2016, 25/03/2016, 28/03/2016
                                <1>     ; 21/03/2016, 22/03/2016, 23/03/2016
                                <1>     ; 16/03/2016, 17/03/2016, 18/03/2016
                                <1>     ; 15/03/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>     ; 02/09/2011 (FILE.ASM 'copy_source_file_to_destination_file')
                                <1>     ; 01/08/2010 - 18/05/2011
                                <1>     ;
                                <1>     ;   Command Interpreter phase 1 enter ->
                                <1>     ;           AL = 1 -> Caller is command interpreter
                                <1>     ;           AL = 2 -> The second call, re-enter/continue
                                <1>     ;   Phase 1 -> Check source file
                                <1>     ;              'found' is required
                                <1>     ;   Phase 2 -> Check destination file,
                                <1>     ;              save 'found' or 'not found' status
                                <1>     ;              'permission denied' error will be return
                                <1>     ;              if attributes have not for ordinary file
                                <1>     ;              without readonly attribute
                                <1>     ;   Command Interpreter phase 1 return ->
                                <1>     ;              DH = Source file attributes
                                <1>     ;              DL = Destination file found status
                                <1>     ;              EAX = 0
                                <1>     ;   Command Interpreter phase 2 enter ->
                                <1>     ;              AL = 2 -> Continue from the last position
                                <1>     ;              AH =
                                <1>     ;   Phase 3 -> Load source file or use read/write cluster method
                                <1>     ;   Phase 4 -> Create destination file if it is not found
                                <1>     ;   Phase 5 -> Open destination file
                                <1>     ;   Phase 6 -> Read from source and write to destination
                                <1>     ;   Phase 7 -> Unload source file, if it is loaded at memory
                                <1>     ;       cf = 1 causes to return before the phase 7
                                <1>     ;              but loaded file will be unloaded
                                <1>     ;             (allocated memory block will be deallocated)
                                <1>     ;
                                <1>     ; INPUT ->
                                <1>     ;       ESI = Source File Pathname (Asciiz)
                                <1>     ;       EDI = Destination File Pathname (Asciiz)
                                <1>     ;       AL = 0 --> Interrupt (System call)
                                <1>     ;       AL > 0 --> Command Interpreter (Question)
                                <1>     ;       AL = 1 --> Question Phase
                                <1>     ;       AL = 2 --> Progress Phase
                                <1>     ;
                                <1>     ; OUTPUT ->
                                <1>     ;      cf = 0 -> OK ; (*)
                                <1>     ;      EAX = Destination file first cluster
                                <1>     ;
                                <1>     ;      31/08/2024 - TRDOS v2.0.9 ; (*)
                                <1>     ; obsolete ;CL > 0 if there is file reading error before EOF
                                <1>     ;          ;           (incomplete copy)
                                <1>     ;          ;CH > 0 if file is (full) loaded at memory
                                <1>     ;
                                <1>     ;      cf = 1 -> Error code in AL (EAX) ; (*)
                                <1>     ;
                                <1>     ; (EBX, ECX, ESI, EDI register contents will be changed)
                                <1>
0000B3ED 3C02                   <1>     cmp     al, 2
                                <1>     ;je     csftdf2_check_cdrv
                                <1>     ; 29/07/2022
0000B3EF 7205                   <1>     jb      short csftdf_ph1
                                <1>     ;jmp    csftdf2_check_cdrv
0000B3F1 E943020000             <1>     jmp     csftdf_ph2
                                <1>
                                <1> ; Phase 1
                                <1>     ; 29/07/2022
                                <1> csftdf_ph1:
0000B3F6 A2[54820100]           <1>     mov     byte [copy_cmd_phase], al
                                <1>
0000B3FB 57                     <1>     push    edi ; *
                                <1>
                                <1> csftdf_parse_sf_path:
0000B3FC BF[2C810100]           <1>     mov     edi, SourceFile_Drv
0000B401 E868F5FFFF             <1>     call    parse_path_name
0000B406 721C                   <1>     jc      short csftdf_parse_sf_path_failed
                                <1>
                                <1> csftdf_parse_df_path:
0000B408 5E                     <1>     pop     esi ; * (pushed edi)
                                <1>
                                <1> csftdf_sf_check_filename_exists:
0000B409 803D[6E810100]21       <1>     cmp     byte [SourceFile_Name], 21h
0000B410 7214                   <1>     jb      short csftdf_sf_file_not_found_error
                                <1>
0000B412 BF[AC810100]           <1>     mov     edi, DestinationFile_Drv
0000B417 E852F5FFFF             <1>     call    parse_path_name
0000B41C 730E                   <1>     jnc     short csftdf_check_sf_cdrv
                                <1>
```

```
                                <1>     cmp     al, 1 ; File or directory name is not existing
0000B41E 3C01                   <1>     jna     short csftdf_check_sf_cdrv
0000B420 760A                   <1>
                                <1> csftdf_parse_df_path_failed:
                                <1>     stc
0000B422 F9                     <1> csftdf_sf_error_retn:
                                <1>     retn
0000B423 C3                     <1>
                                <1> csftdf_parse_sf_path_failed:
                                <1>     pop     edi ; *
0000B424 5F                     <1>     ;jmp    short csftdf_sf_error_retn
                                <1>     ; 25/08/2024
0000B425 C3                     <1>     retn
                                <1>
                                <1> csftdf_sf_file_not_found_error:
0000B426 B802000000             <1>     mov     eax, 2 ; File not found
                                <1>     ;jmp    short csftdf_sf_error_retn
                                <1>     ; 25/08/2024
0000B42B C3                     <1>     retn
                                <1>
                                <1> csftdf_check_sf_cdrv:
0000B42C 8A3D[42770100]         <1>     mov     bh, [Current_Drv]
                                <1>
0000B432 883D[57820100]         <1>     mov     [csftdf_cdrv], bh ; 23/03/2016
                                <1>
0000B438 8A15[2C810100]         <1>     mov     dl, [SourceFile_Drv]
0000B43E 38FA                   <1>     cmp     dl, bh ; byte [Current_Drv]
0000B440 7407                   <1>     je      short csftdf_sf_check_directory
                                <1>
0000B442 E899C2FFFF             <1>     call    change_current_drive
0000B447 72DA                   <1>     jc      short csftdf_sf_error_retn
                                <1>
                                <1> csftdf_sf_check_directory:
0000B449 BE[2D810100]           <1>     mov     esi, SourceFile_Directory
0000B44E 803E20                 <1>     cmp     byte [esi], 20h
0000B451 760F                   <1>     jna     short csftdf_find_sf
                                <1>
                                <1> csftdf_sf_change_directory:
0000B453 FE05[5D2E0100]         <1>     inc     byte [Restore_CDIR]
0000B459 30E4                   <1>     xor     ah, ah ; CD_COMMAND sign -> 0
0000B45B E87BEFFFFF             <1>     call    change_current_directory
0000B460 72C1                   <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:
0000B462 BE[6E810100]           <1>     mov     esi, SourceFile_Name
0000B467 66B80018               <1>     mov     ax, 1800h ; Except volume label and dirs
0000B46B E8FAD4FFFF             <1>     call    find_first_file
0000B470 72B1                   <1>     jc      short csftdf_sf_error_retn
                                <1>
                                <1> csftdf_sf_ambgfn_check:
0000B472 6621D2                 <1>     and     dx, dx ; Ambiguous filename chars used sign (DX>0)
0000B475 7406                   <1>     jz      short csftdf_sf_found
                                <1>
                                <1> csftdf_ambiguous_file_name_error:
                                <1>     ;mov    eax, 2 ; File not found error
                                <1>     ; 29/07/2022
0000B477 29C0                   <1>     sub     eax, eax
0000B479 B002                   <1>     mov     al, 2
0000B47B F9                     <1>     stc
0000B47C C3                     <1>     retn
                                <1>
                                <1> csftdf_sf_found:
0000B47D A3[58820100]           <1>     mov     [csftdf_filesize], eax
                                <1>
0000B482 09C0                   <1>     or      eax, eax
0000B484 7504                   <1>     jnz     short csftdf_set_source_file_direntry
                                <1>
                                <1> ;csftdf_sf_file_size_zero:
                                <1>     ;mov    eax, 20 ; TRDOS zero length (file size) error
                                <1>     ; 25/08/2024
                                <1>     ; eax = 0
                                <1>     ; 29/07/2022
                                <1>     ;sub    eax, eax
0000B486 B014                   <1>     mov     al, 20
0000B488 F9                     <1>     stc
0000B489 C3                     <1>     retn
                                <1>
                                <1> csftdf_set_source_file_direntry:
0000B48A BE[34800100]           <1>     mov     esi, FindFile_DirEntry
0000B48F BF[7E810100]           <1>     mov     edi, SourceFile_DirEntry
                                <1>     ;mov    ecx, 8
                                <1>     ; 29/07/2022
0000B494 31C9                   <1>     xor     ecx, ecx
0000B496 B108                   <1>     mov     cl, 8
0000B498 F3A5                   <1>     rep     movsd
                                <1>
                                <1> csftdf_sf_restore_cdrv:
                                <1>     ; 22/03/2016
0000B49A 8A15[57820100]         <1>     mov     dl, [csftdf_cdrv]
0000B4A0 3A15[42770100]         <1>     cmp     dl, [Current_Drv]
0000B4A6 7407                   <1>     je      short csftdf_sf_restore_cdir
0000B4A8 E833C2FFFF             <1>     call    change_current_drive
0000B4AD 724F                   <1>     jc      short csftdf_df_error_retn ; 30/03/2016
                                <1>
                                <1> csftdf_sf_restore_cdir:
0000B4AF 803D[5D2E0100]00       <1>     cmp     byte [Restore_CDIR], 0
0000B4B6 7612                   <1>     jna     short csftdf_df_check_filename_exists
0000B4B8 29C0                   <1>     sub     eax, eax
0000B4BA BE00010900             <1>     mov     esi, Logical_DOSDisks
0000B4BF 88D4                   <1>     mov     ah, dl ; byte [csftdf_cdrv]
0000B4C1 01C6                   <1>     add     esi, eax
0000B4C3 E8C9C2FFFF             <1>     call    restore_current_directory
0000B4C8 7234                   <1>     jc      short csftdf_df_error_retn
                                <1>
                                <1> csftdf_df_check_filename_exists:
0000B4CA 803D[EE810100]20       <1>     cmp     byte [DestinationFile_Name], 20h
0000B4D1 7716                   <1>     ja      short csftdf_check_df_cdrv
                                <1>
                                <1> csftdf_copy_sf_name:
0000B4D3 BF[EE810100]           <1>     mov     edi, DestinationFile_Name
0000B4D8 BE[6E810100]           <1>     mov     esi, SourceFile_Name
0000B4DD B10C                   <1>     mov     cl, 12
                                <1>
                                <1> csftdf_df_copy_sf_name_loop:
0000B4DF AC                     <1>     lodsb
0000B4E0 AA                     <1>     stosb
0000B4E1 08C0                   <1>     or      al, al
0000B4E3 7404                   <1>     jz      short csftdf_check_df_cdrv
0000B4E5 FEC9                   <1>     dec     cl
0000B4E7 75F6                   <1>     jnz     csftdf_df_copy_sf_name_loop
                                <1>
```

```
                                <1> csftdf_check_df_cdrv:
0000B4E9 8A15[AC810100]         <1>     mov     dl, [DestinationFile_Drv]
0000B4EF 3A15[42770100]         <1>     cmp     dl, [Current_Drv]
0000B4F5 7408                   <1>     je      short csftdf_df_check_directory
                                <1>
0000B4F7 E8E4C1FFFF             <1>     call    change_current_drive
0000B4FC 7301                   <1>     jnc     short csftdf_df_check_directory
                                <1>
                                <1> csftdf_df_error_retn:
0000B4FE C3                     <1>     retn
                                <1>
                                <1> csftdf_df_check_directory:
0000B4FF BE[AD810100]           <1>     mov     esi, DestinationFile_Directory
0000B504 803E20                 <1>         cmp     byte [esi], 20h
0000B507 760F                   <1>     jna     short csftdf_find_df
                                <1>
                                <1> csftdf_df_change_directory:
0000B509 FE05[5D2E0100]         <1>     inc     byte [Restore_CDIR]
0000B50F 28E4                   <1>     sub     ah, ah ; CD_COMMAND sign -> 0
0000B511 E8C5EEFFFF             <1>     call    change_current_directory
0000B516 72E6                   <1>     jc      short csftdf_df_error_retn
                                <1>
                                <1> ;csftdf_df_change_prompt_dir_string:
                                <1> ;   call    change_prompt_dir_string
                                <1>
                                <1> csftdf_find_df:
                                <1>     ; 23/03/2016
0000B518 29DB                   <1>     sub     ebx, ebx
0000B51A 8A3D[AC810100]         <1>     mov     bh, [DestinationFile_Drv]
0000B520 81C300010900           <1>     add     ebx, Logical_DOSDisks
0000B526 891D[84820100]         <1>     mov     [csftdf_df_drv_dt], ebx
                                <1>
0000B52C BE[EE810100]           <1>     mov     esi, DestinationFile_Name
                                <1>     ;xor    ax, ax
                                <1>     ; 25/08/2024
0000B531 31C0                   <1>     xor     eax, eax
                                <1>             ; DestinationFile_AttributesMask -> any/zero
0000B533 E832D4FFFF             <1>     call    find_first_file
0000B538 7217                   <1>     jc      short csftdf_df_check_error_code
                                <1>
                                <1> csftdf_df_ambgfn_check:
0000B53A 6609D2                 <1>     or      dx, dx ; Ambiguous filename chars used sign (DX>0)
0000B53D 7529                   <1>     jnz     short csftdf_df_error_inv_fname
                                <1>
                                <1> csftdf_df_found:
0000B53F C605[56820100]01       <1>     mov     byte [DestinationFileFound], 1
                                <1>     ; 17/10/2016 (cl -> bl)
0000B546 80E31F                 <1>     and     bl, 1Fh ; Attributes, D-V-S-H-R
0000B549 7452                   <1>     jz      short csftdf_df_save_first_cluster
                                <1>
                                <1> csftdf_df_permission_denied_retn:
                                <1>     ;mov    eax, 05h ; Access/Permission denied.
                                <1>     ; 29/07/2022
0000B54B 29C0                   <1>     sub     eax, eax
0000B54D B005                   <1>     mov     al, 5
                                <1> csftdf_df_error_stc_retn:
0000B54F F9                     <1>     stc
0000B550 C3                     <1>     retn
                                <1>
                                <1> csftdf_df_check_error_code:
                                <1>     ;cmp    eax, 2
0000B551 3C02                   <1>     cmp     al, 2
0000B553 75FA                   <1>     jne     short csftdf_df_error_stc_retn
                                <1>
0000B555 C605[56820100]00       <1>     mov     byte [DestinationFileFound], 0
                                <1>
                                <1>     ; 15/10/2016
0000B55C BE[24800100]           <1>     mov     esi, FindFile_Name ; *
0000B561 E863D7FFFF             <1>     call    check_filename
0000B566 7306                   <1>     jnc     short csftdf_df_valid_fname
                                <1> csftdf_df_error_inv_fname: ; 'invalid file name !'
                                <1>     ;mov    eax, ERR_INV_FILE_NAME  ; 26
                                <1>     ; 29/07/2022
0000B568 29C0                   <1>     sub     eax, eax
0000B56A B01A                   <1>     mov     al, ERR_INV_FILE_NAME ; 26
0000B56C F9                     <1>     stc
0000B56D C3                     <1>     retn
                                <1>
                                <1> csftdf_df_valid_fname:
                                <1>     ; 21/03/2016
                                <1>     ; (Capitalized file name)
                                <1>     ;mov    esi, FindFile_Name ; * ; 15/10/2016
0000B56E BF[EE810100]           <1>     mov     edi, DestinationFile_Name
0000B573 A5                     <1>     movsd
0000B574 A5                     <1>     movsd
0000B575 A5                     <1>     movsd
                                <1>     ;movsb
                                <1>
                                <1> csftdf_check_disk_free_size_0:
0000B576 A1[9A810100]           <1>     mov     eax, [SourceFile_DirEntry+DirEntry_FileSize]
                                <1>
                                <1> csftdf_check_disk_free_size_1:
                                <1>     ;sub    ebx, ebx
                                <1>     ;mov    esi, Logical_DOSDisks
                                <1>     ;mov    bh,  [DestinationFile_Drv]
                                <1>     ;add    esi, ebx
                                <1>
0000B57B 8B35[84820100]         <1>     mov     esi, [csftdf_df_drv_dt] ; 23/03/2016
                                <1>
                                <1>     ;movzx  ecx, word [esi+LD_BPB+BytesPerSec] ; 17, LD_BPB + 0Bh
                                <1>     ; 19/12/2025
0000B581 B900020000             <1>     mov     ecx, 512
0000B586 01C8                   <1>     add     eax, ecx
0000B588 48                     <1>     dec     eax  ; file size (additional bytes) + 511 (round up)
                                <1> csftdf_check_disk_free_size_3: ; 16/03/2016
0000B589 29D2                   <1>     sub     edx, edx
0000B58B F7F1                   <1>     div     ecx ; bytes per sector
                                <1>
                                <1> csftdf_check_disk_free_size:
0000B58D 3B4674                 <1>     cmp     eax, [esi+LD_FreeSectors]
                                <1>         ;jb csftdf_check_disk_free_size_ok
                                <1>     ; 25/08/2024
                                <1>     ; 29/07/2022
                                <1>     ;jb     short csftdf_check_dfs_ok
0000B590 7705                   <1>     ja      short csftdf_df_insufficient_disk_space
                                <1>
                                <1>     ; 25/08/2024
                                <1>     ;cmp    byte [esi+LD_FATType], 0 ; FS needs FDT sector also.
                                <1>     ;;ja    csftdf_check_disk_free_size_ok
                                <1>     ;; 29/07/2022
                                <1>     ;jna    short csftdf_df_insufficient_disk_space
                                <1>
                                <1> csftdf_check_dfs_ok:
0000B592 E985000000             <1>     jmp     csftdf_check_disk_free_size_ok
```

```
                                <1>
                                <1> csftdf_df_insufficient_disk_space:
                                <1>     ;mov    eax, 27h ; insufficient disk space
                                <1>     ; 29/07/2022
0000B597 29C0                   <1>     sub     eax, eax
0000B599 B027                   <1>     mov     al, 27h
                                <1>     ;jmp    short csftdf_df_error_stc_retn
                                <1>     ; 25/08/2024
0000B59B F9                     <1>     stc
0000B59C C3                     <1>     retn
                                <1>
                                <1> csftdf_df_save_first_cluster:
                                <1>     ; ESI = FindFile_DirEntry (for the old destination file)
                                <1>     ; EAX = Old destination file size
                                <1>     ; 24/03/2016
                                <1>     ; EDI = Directory entry address (within Dir Buffer boundaries)
0000B59D 81EF00000800           <1>     sub     edi, Directory_Buffer  ; (<65536)
                                <1>     ;shr    di, 5 ; Convert entry offset to entry index/number
                                <1>     ; 29/07/2022
0000B5A3 C1EF05                 <1>     shr     edi, 5
0000B5A6 66893D[26820100]       <1>     mov     [DestinationFile_DirEntryNumber], di ; (<2048)
                                <1>
                                <1> csftdf_df_check_sf_df_fcluster:
0000B5AD 668B5614               <1>     mov     dx, [esi+DirEntry_FstClusHI]
0000B5B1 C1E210                 <1>     shl     edx, 16
0000B5B4 668B561A               <1>     mov     dx, [esi+DirEntry_FstClusLO]
0000B5B8 8915[68820100]         <1>     mov     [csftdf_df_cluster], edx
                                <1> csftdf_df_check_sf_df_fcluster_1:
0000B5BE 668B15[92810100]       <1>     mov     dx, [SourceFile_DirEntry+DirEntry_FstClusHI]
0000B5C5 C1E210                 <1>     shl     edx, 16
0000B5C8 668B15[98810100]       <1>     mov     dx, [SourceFile_DirEntry+DirEntry_FstClusLO]
0000B5CF 3B15[68820100]         <1>     cmp     edx, [csftdf_df_cluster]
0000B5D5 7512                   <1>     jne     short csftdf_df_check_sf_df_fcluster_ok
                                <1> csftdf_df_check_sf_df_drv:
0000B5D7 8A15[2C810100]         <1>     mov     dl, [SourceFile_Drv]
0000B5DD 3A15[AC810100]         <1>     cmp     dl, [DestinationFile_Drv]
0000B5E3 7504                   <1>     jne     short csftdf_df_check_sf_df_fcluster_ok
                                <1>
                                <1>     ; source and destination files are same !
                                <1>     ; (they have same first cluster value on same logical disk)
                                <1>
0000B5E5 31C0                   <1>     xor     eax, eax ; mov eax, 0 -> Bad command or file name !
0000B5E7 F9                     <1>     stc
0000B5E8 C3                     <1>     retn
                                <1>
                                <1> csftdf_df_check_sf_df_fcluster_ok:
                                <1> csftdf_df_move_findfile_struct:
                                <1>     ; mov   esi, FindFile_DirEntry
0000B5E9 BF[FE810100]           <1>     mov     edi, DestinationFile_DirEntry
                                <1>     ;mov    ecx, 8
0000B5EE 31C9                   <1>     xor     ecx, ecx
0000B5F0 B108                   <1>     mov     cl, 8
0000B5F2 F3A5                   <1>     rep     movsd
                                <1>
                                <1> ;csftdf_check_disk_free_size_2:
0000B5F4 89C2                   <1>     mov     edx, eax ; Old destination file size
                                <1>
                                <1>     ;mov    eax, [SourceFile_DirEntry+DirEntry_FileSize]
0000B5F6 A1[58820100]           <1>     mov     eax, [csftdf_filesize] ; 23/03/2016
                                <1>
                                <1>     ;;sub   ecx, ecx ; 0
                                <1>     ;mov    esi, Logical_DOSDisks
                                <1>     ;mov    ch, [DestinationFile_Drv]
                                <1>     ;add    esi, ecx
                                <1>     ;
                                <1>     ;mov    [csftdf_df_drv_dt], esi
                                <1>
0000B5FB 8B35[84820100]         <1>     mov     esi, [csftdf_df_drv_dt] ; 23/03/2016
                                <1>
                                <1>     ;mov    cx, [esi+LD_BPB+BytesPerSec] ; 17, LD_BPB + 0Bh
                                <1>     ; 19/12/2025
0000B601 66B90002               <1>     mov     cx, 512
0000B605 01CA                   <1>     add     edx, ecx ; + 512
0000B607 01C8                   <1>     add     eax, ecx ; + 512
0000B609 4A                     <1>     dec     edx ; old file size + 511 (round up)
0000B60A 48                     <1>     dec     eax ; new file size + 511 (round up)
0000B60B F7D9                   <1>     neg     ecx ; -512 ; 0FFFFFE00h
0000B60D 21CA                   <1>     and     edx, ecx ; = old sector count * 512
0000B60F 21C8                   <1>     and     eax, ecx ; = new sector count * 512
                                <1>
0000B611 29D0                   <1>     sub     eax, edx ; new file size - old file size (on disk)
0000B613 7607                   <1>     jna     short csftdf_check_disk_free_size_ok
                                <1>
0000B615 F7D9                   <1>     neg     ecx ; 512 (bytes per sector) ; 200h
                                <1>     ; check free space for additional sectors
                                <1>     ; eax = number of additional sectors * bytes per sector
                                <1>     ; esi = Logical DOS drive number (of destination disk)
0000B617 E96DFFFFFF             <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:
0000B61C A0[54820100]           <1>     mov     al, [copy_cmd_phase]
0000B621 3C01                   <1>     cmp     al, 1
0000B623 7514                   <1>     jne     short csftdf2_check_cdrv
                                <1>
0000B625 31C0                   <1>     xor     eax, eax
0000B627 A2[54820100]           <1>     mov     [copy_cmd_phase], al ; 0
                                <1>
0000B62C 8A15[56820100]         <1>     mov     dl, [DestinationFileFound]
0000B632 8A35[89810100]         <1>     mov     dh, [SourceFile_DirEntry+11] ; Attributes
                                <1>
                                <1> csftdf_return:
0000B638 C3                     <1>     retn
                                <1>
                                <1> ; Phase 2
                                <1> csftdf_ph2:
                                <1>     ; 29/07/2022
                                <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
0000B639 803D[56820100]00       <1>     cmp     byte [DestinationFileFound], 0
0000B640 7742                   <1>     ja      short csftdf2_set_sf_percentage
                                <1>
                                <1> csftdf2_create_file:
0000B642 BE[EE810100]           <1>     mov     esi, DestinationFile_Name
0000B647 A1[58820100]           <1>     mov     eax, [csftdf_filesize]
0000B64C 30C9                   <1>     xor     cl, cl ; 0
                                <1>
0000B64E 31DB                   <1>     xor     ebx, ebx ; 0
0000B650 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>     ;                 (0 for empty file) ; 29/08/2024
                                <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>     ;       ; 29/08/2024
                                <1>     ;       EBX = File Size (0 for a new, empty file)
                                <1>     ;       ECX = Directory Entry Index/Number (<2048)
                                <1>     ;             (in directory cluster, not in directory)
                                <1>     ;       EDX = Directory Cluster Number (of the file)
                                <1>     ;
                                <1>     ;       cf = 1 -> error code in AL (EAX)
                                <1>
0000B651 E80D060000             <1>     call    create_file
                                <1>     ;pop    esi
                                <1>     ;jc     csftdf2_rw_error
                                <1>     ; 29/07/2022
0000B656 7305                   <1>     jnc     short csftdf2_create_file_OK
0000B658 E978050000             <1>     jmp     csftdf2_rw_error
                                <1>
                                <1> csftdf2_create_file_OK:
                                <1>
0000B65D A3[68820100]           <1>     mov     [csftdf_df_cluster], eax
                                <1>     ; 27/08/2024
                                <1>     ; eax = 0
                                <1>
                                <1>     ; 24/03/2016
                                <1>     ;mov    [DestinationFile_DirEntryNumber], dx
                                <1>     ; 29/08/2024
                                <1>     ; (these 3 parameters are needed for
                                <1>     ;  loading same dir entry for first cluster update)
0000B662 8915[22820100]         <1>     mov     [DestinationFile_DirCluster], edx
0000B668 66890D[26820100]       <1>     mov     [DestinationFile_DirEntryNumber], cx
                                <1>
                                <1>     ; 30/08/2024
                                <1>     ; edx -> ecx
                                <1>     ; 21/03/2016
0000B66F BE00000800             <1>     mov     esi, Directory_Buffer
                                <1>     ;shl    edx, 5 ; 32 * index number
0000B674 C1E105                 <1>     shl     ecx, 5
                                <1>     ;add    esi, edx
0000B677 01CE                   <1>     add     esi, ecx
0000B679 BF[FE810100]           <1>     mov     edi, DestinationFile_DirEntry
                                <1>     ;mov    cl, 8 ; 32 bytes
                                <1>     ; 29/08/2024
0000B67E 29C9                   <1>     sub     ecx, ecx
0000B680 B108                   <1>     mov     cl,8
0000B682 F3A5                   <1>     rep     movsd
                                <1>
                                <1>     ;mov    cl, [esi] ; L.D.D.D.T.
                                <1>     ;mov    [DestinationFile_Drv], cl
                                <1>
                                <1> csftdf2_set_sf_percentage:
                                <1>     ; 17/03/2016
0000B684 31C0                   <1>     xor     eax, eax
0000B686 A2[7C820100]           <1>     mov     [csftdf_percentage], al ; 0, reset
                                <1>
0000B68B A3[74820100]           <1>     mov     [csftdf_sf_rbytes], eax ; 0, reset
0000B690 A3[78820100]           <1>     mov     [csftdf_df_wbytes], eax ; 0, reset
                                <1>
0000B695 8A25[2C810100]         <1>     mov     ah, [SourceFile_Drv]
0000B69B BE00010900             <1>     mov     esi, Logical_DOSDisks
0000B6A0 01C6                   <1>     add     esi, eax
                                <1>
0000B6A2 8935[80820100]         <1>     mov     [csftdf_sf_drv_dt], esi ; 23/03/2016
                                <1>
0000B6A8 668B15[92810100]       <1>     mov     dx, [SourceFile_DirEntry+DirEntry_FstClusHI]
0000B6AF C1E210                 <1>     shl     edx, 16
0000B6B2 668B15[98810100]       <1>     mov     dx, [SourceFile_DirEntry+DirEntry_FstClusLO]
0000B6B9 8915[64820100]         <1>     mov     [csftdf_sf_cluster], edx
                                <1>
                                <1>     ; 16/03/2016
                                <1>     ; Note: Singlix FS boot sector parameters (for cluster
                                <1>     ;       related calculations) has same offset
                                <1>     ;       values from LD_BPB as in FAT file system.
                                <1>     ;       [esi+LD_BPB+SecPerClust] is 1 for Singlix FS.
                                <1>     ;
0000B6BF 0FB64E13               <1>     movzx   ecx, byte [esi+LD_BPB+SecPerClust]
0000B6C3 880D[AA810100]         <1>     mov     [SourceFile_SecPerClust], cl
                                <1>
                                <1> ; 19/12/2025
                                <1> %if 0
                                <1>     ; 17/03/2016
                                <1>     cmp     [esi+LD_FATType], ch ; 0
                                <1>     ja      short csftdf2_set_sf_percent_rsize1
                                <1>
                                <1>     mov     eax, 65536 ; read/write buffer size for Singlix FS
                                <1>     jmp     short csftdf2_set_sf_percent_rsize2
                                <1> %endif
                                <1>
                                <1> csftdf2_set_sf_percent_rsize1:
```

```
                            <1>     ;mov    ax, [esi+LD_BPB+BytesPerSec]
                            <1>     ; 19/12/2025
0000B6C9 66B80002           <1>     mov     ax, 512
0000B6CD F7E1               <1>     mul     ecx
                            <1>     ;sub    edx, edx
                            <1> csftdf2_set_sf_percent_rsize2:
0000B6CF A3[6C820100]       <1>     mov     [csftdf_r_size], eax
                            <1>
                            <1> csftdf2_set_df_percentage:
                            <1>     ;sub    eax, eax
                            <1>     ;mov    ah, [DestinationFile_Drv]
                            <1>     ;mov    edi, Logical_DOSDisks
                            <1>     ;add    edi, eax
                            <1>     ;mov    [csftdf_df_drv_dt], edi ; 17/03/2016
                            <1>
0000B6D4 8B3D[84820100]     <1>     mov     edi, [csftdf_df_drv_dt] ; 23/03/2016
                            <1>
                            <1>     ; 16/03/2016
                            <1>     ; Note: Singlix FS boot sector parameters (for cluster
                            <1>     ;       related calculations) has same offset
                            <1>     ;       values from LD_BPB as in FAT file system.
                            <1>     ;       [edi+LD_BPB+SecPerClust] is 1 for Singlix FS.
                            <1>     ;
                            <1>     ;movzx  ecx, byte [edi+LD_BPB+SecPerClust]
0000B6DA 8A4F13             <1>     mov     cl, [edi+LD_BPB+SecPerClust]
0000B6DD 880D[2A820100]     <1>     mov     [DestinationFile_SecPerClust], cl
                            <1>
                            <1> ; 19/12/2025
                            <1> %if 0
                            <1>     ; 17/03/2016
                            <1>     cmp     [edi+LD_FATType], ch ; 0
                            <1>     ja      short csftdf2_set_df_percent_wsize1
                            <1>
                            <1>     mov     eax, 65536 ; read/write buffer size for Singlix FS
                            <1>     jmp     short csftdf2_set_df_percent_wsize2
                            <1> %endif
                            <1>
                            <1> csftdf2_set_df_percent_wsize1:
                            <1>     ;movzx  eax, word [edi+LD_BPB+BytesPerSec]
                            <1>     ; 19/12/2025
0000B6E3 B800020000         <1>     mov     eax, 512
0000B6E8 F7E1               <1>     mul     ecx
                            <1>     ;sub    edx, edx
                            <1> csftdf2_set_df_percent_wsize2:
0000B6EA A3[70820100]       <1>     mov     [csftdf_w_size], eax
                            <1>
0000B6EF A1[58820100]       <1>     mov     eax, [csftdf_filesize]
                            <1>
0000B6F4 3D00000100         <1>     cmp     eax, 65536 ; 64KB     ; small file
0000B6F9 721F               <1>     jb      short csftdf2_load_file ; do not display percentage
                            <1>
                            <1> csftdf2_reset_wf_percent_ptr_chk_64k:
0000B6FB B201               <1>     mov     dl, 1 ; 25/03/2016
                            <1>
0000B6FD 3D00000400         <1>     cmp     eax, 65536*4 ; 256KB
0000B702 7310               <1>     jnb     short csftdf2_enable_percentage_display ; big file
                            <1>
                            <1>     ; 64-128KB file size for floppy disks
0000B704 3815[2C810100]     <1>     cmp     byte [SourceFile_Drv], dl ; 1 ; read from floppy disk ?
0000B70A 7608               <1>     jna     short csftdf2_enable_percentage_display
                            <1>
0000B70C 3815[AC810100]     <1>     cmp     byte [DestinationFile_Drv], dl ; 1 ; write to floppy disk ?
0000B712 7706               <1>     ja      short csftdf2_load_file
                            <1>
                            <1> csftdf2_enable_percentage_display:
0000B714 8815[7C820100]     <1>     mov     [csftdf_percentage], dl ; 1
                            <1>
                            <1> csftdf2_load_file:
                            <1>     ; 13/05/2016
                            <1>     ; 19/03/2016
                            <1>     ; 18/03/2016
                            <1>     ; 17/03/2016
0000B71A B40F               <1>     mov     ah, 0Fh
0000B71C E80360FFFF         <1>     call    _int10h
                            <1>     ; 13/05/2016
0000B721 883D[7D820100]     <1>     mov     [csftdf_videopage], bh ; active video page
0000B727 B403               <1>     mov     ah, 03h
0000B729 E8F65FFFFF         <1>     call    _int10h
0000B72E 668915[7E820100]   <1>     mov     [csftdf_cursorpos], dx
                            <1>
0000B735 29C0               <1>     sub     eax, eax
0000B737 A2[55820100]       <1>     mov     [csftdf_rw_err], al ; 0
                            <1>
                            <1> ; ///
                            <1> csftdf_sf_amb: ; 15/03/2016
0000B73C 8B0D[58820100]     <1>     mov     ecx, [csftdf_filesize]; 23/03/2016
                            <1>
                            <1>     ; TRDOS 386 (TRDOS v2.0)
                            <1>     ; Allocate contiguous memory block for loading the file
                            <1>
                            <1>     ;mov    ecx, [SourceFile_DirEntry+DirEntry_FileSize]
                            <1>
                            <1>     ;sub    eax, eax ; First free memory aperture
                            <1>
                            <1>     ; eax = 0 (Allocate memory from the beginning)
                            <1>     ; ecx = File (Allocation) size in bytes
                            <1>
                            <1> ; 31/08/2024 - temporary
                            <1> ;%if 0
0000B742 E8F2A5FFFF         <1>     call    allocate_memory_block
0000B747 7304               <1>     jnc     short loc_check_sf_save_loading_parms
                            <1> ;%endif
                            <1>
                            <1> csftdf_use_cluster_buff: ; 27/08/2024
                            <1>
0000B749 29C0               <1>     sub     eax, eax
0000B74B 29C9               <1>     sub     ecx, ecx
                            <1>
                            <1> loc_check_sf_save_loading_parms:
0000B74D A3[5C820100]       <1>     mov     [csftdf_sf_mem_addr], eax ; loading address
0000B752 890D[60820100]     <1>     mov     [csftdf_sf_mem_bsize], ecx ; block size
                            <1> ; ///
                            <1>     ; 19/03/2016
0000B758 8B35[80820100]     <1>     mov     esi, [csftdf_sf_drv_dt] ; logical dos drv desc. tbl.
                            <1>
                            <1>     ; 17/03/2016
0000B75E 09C0               <1>     or      eax, eax ; contiguous free memory block address
                            <1>     ;jz     csftdf2_read_sf_cluster
                            <1>     ; 29/07/2022
0000B760 7505               <1>     jnz     short csftdf2_1
0000B762 E949010000         <1>     jmp     csftdf2_read_sf_cluster
                            <1>
                            <1> csftdf2_1:
                            <1>     ; 18/03/2016
```

```
4198 0000B767 8B1D[5C820100]      <1>     mov     ebx, [csftdf_sf_mem_addr] ; memory block address
4199                              <1>
4200                              <1> ; 19/12/2025
4201                              <1> %if 0
4202                              <1>     cmp     byte [esi+LD_FATType], 0
4203                              <1>         ;jna       csftdf2_load_fs_file
4204                              <1>     ; 29/07/2022
4205                              <1>     ja      short csftdf2_load_fat_file
4206                              <1>     jmp     csftdf2_load_fs_file
4207                              <1> %endif
4208                              <1>
4209                              <1> csftdf2_load_fat_file:
4210 0000B76D 53                  <1>     push    ebx ; *
4211                              <1>
4212                              <1> csftdf2_load_fat_file_next:
4213 0000B76E BE[B5340100]        <1>     mov     esi, msg_reading
4214 0000B773 E832B6FFFF          <1>     call    print_msg
4215                              <1>
4216 0000B778 803D[7C820100]00    <1>     cmp     byte [csftdf_percentage], 0
4217 0000B77F 7605                <1>     jna     short csftdf2_load_fat_file_1
4218                              <1>
4219 0000B781 E87E000000          <1>     call    csftdf2_print_percentage ; 19/03/2016
4220                              <1>
4221                              <1> csftdf2_load_fat_file_1:
4222 0000B786 8B35[80820100]      <1>     mov     esi, [csftdf_sf_drv_dt]
4223 0000B78C 5B                  <1>     pop     ebx ; *
4224                              <1>
4225                              <1> csftdf2_load_fat_file_2:
4226 0000B78D E8AF000000          <1>     call    csftdf2_read_fat_file_sectors ; 19/03/2016
4227                              <1>     ;jc     csftdf2_rw_error ; eocc! or disk error!
4228                              <1>     ; 29/07/2022
4229 0000B792 7305                <1>     jnc     short csftdf2_load_fat_file_3
4230 0000B794 E93C040000          <1>     jmp     csftdf2_rw_error
4231                              <1> csftdf2_load_fat_file_3:
4232 0000B799 09D2                <1>     or      edx, edx ; edx > 0 -> EOF
4233 0000B79B 7520                <1>     jnz     short csftdf2_load_fat_file_ok
4234                              <1>
4235 0000B79D 803D[7C820100]00    <1>     cmp     byte [csftdf_percentage], 0
4236 0000B7A4 76E7                <1>     jna     short csftdf2_load_fat_file_2
4237                              <1>
4238 0000B7A6 53                  <1>     push    ebx ; *
4239                              <1>
4240                              <1>     ; Set cursor position
4241                              <1>     ; AH= 02h, BH= Page Number, DH= Row, DL= Column
4242 0000B7A7 8A3D[7D820100]      <1>     mov     bh, [csftdf_videopage]
4243 0000B7AD 668B15[7E820100]    <1>     mov     dx, [csftdf_cursorpos]
4244 0000B7B4 B402                <1>     mov     ah, 2
4245 0000B7B6 E8695FFFFF          <1>     call    _int10h
4246 0000B7BB EBB1                <1>     jmp     short csftdf2_load_fat_file_next
4247                              <1>
4248                              <1> csftdf2_load_fat_file_ok:
4249 0000B7BD 803D[7C820100]00    <1>     cmp     byte [csftdf_percentage], 0
4250                              <1>     ;jna    csftdf2_save_file ; 25/03/2016
4251                              <1>     ; 29/07/2022
4252 0000B7C4 7705                <1>     ja      short csftdf2_2
4253 0000B7C6 E941020000          <1>     jmp     csftdf2_save_file
4254                              <1> csftdf2_2:
4255                              <1>     ; "Reading... 100%"
4256 0000B7CB BF[CD340100]        <1>     mov     edi, percentagestr
4257 0000B7D0 B031                <1>     mov     al, '1'
4258 0000B7D2 AA                  <1>     stosb
4259 0000B7D3 B030                <1>     mov     al, '0'
4260 0000B7D5 AA                  <1>     stosb
4261 0000B7D6 AA                  <1>     stosb
4262                              <1>
4263 0000B7D7 8A3D[7D820100]      <1>     mov     bh, [csftdf_videopage]
4264 0000B7DD 668B15[7E820100]    <1>     mov     dx, [csftdf_cursorpos]
4265 0000B7E4 B402                <1>     mov     ah, 2
4266 0000B7E6 E8395FFFFF          <1>     call    _int10h
4267                              <1>
4268 0000B7EB BE[B5340100]        <1>     mov     esi, msg_reading
4269 0000B7F0 E8B5B5FFFF          <1>     call    print_msg
4270                              <1>
4271 0000B7F5 BE[CD340100]        <1>     mov     esi, percentagestr
4272 0000B7FA E8ABB5FFFF          <1>     call    print_msg
4273                              <1>
4274 0000B7FF E908020000          <1>     jmp     csftdf2_save_file ; 25/03/2016
4275                              <1>
4276                              <1> csftdf2_print_percentage:
4277                              <1>     ; 09/12/2017
4278                              <1>     ; 19/03/2016
4279                              <1>     ; 18/03/2016
4280 0000B804 B020                <1>     mov     al, 20h
4281 0000B806 BF[CD340100]        <1>     mov     edi, percentagestr
4282 0000B80B AA                  <1>     stosb
4283 0000B80C AA                  <1>     stosb
4284 0000B80D A1[74820100]        <1>     mov     eax, [csftdf_sf_rbytes]
4285                              <1>     ;mov    edx, 100
4286                              <1>     ; 29/07/2022
4287 0000B812 29D2                <1>     sub     edx, edx
4288 0000B814 B264                <1>     mov     dl, 100
4289 0000B816 F7E2                <1>     mul     edx
4290 0000B818 8B0D[58820100]      <1>     mov     ecx, [csftdf_filesize]
4291 0000B81E F7F1                <1>     div     ecx
4292 0000B820 B10A                <1>     mov     cl, 10
4293 0000B822 F6F1                <1>     div     cl
4294 0000B824 80C430              <1>     add     ah, '0'
4295 0000B827 8827                <1>     mov     [edi], ah
4296 0000B829 20C0                <1>     and     al, al
4297 0000B82B 740A                <1>     jz      short csftdf2_print_percent_1
4298 0000B82D 4F                  <1>     dec     edi
4299                              <1>     ;cbw
4300 0000B82E 28E4                <1>     sub     ah, ah ; 09/12/2017
4301 0000B830 F6F1                <1>     div     cl
4302 0000B832 80C430              <1>     add     ah, '0'
4303 0000B835 8827                <1>     mov     [edi], ah
4304                              <1>     ;and    al, al
4305                              <1>     ;jz     short csftdf2_print_percent_1
4306                              <1>     ;dec    edi
4307                              <1>     ;mov    [edi], '1' ; 100%
4308                              <1>
4309                              <1> csftdf2_print_percent_1:
4310 0000B837 BE[CD340100]        <1>     mov     esi, percentagestr
4311                              <1>     ;call   print_msg
4312                              <1>     ;retn
4313 0000B83C E969B5FFFF          <1>     jmp     print_msg
4314                              <1>
4315                              <1> csftdf2_read_file_sectors:
4316                              <1>
4317                              <1> ; 19/12/2025
4318                              <1> %if 0
4319                              <1>     ; 19/03/2016
4320                              <1>     cmp     byte [esi+LD_FATType], 0
4321                              <1>     ;jna    csftdf2_read_fs_file_sectors
```

```
                            <1>     ; 29/07/2022
                            <1>     ja      short csftdf2_read_fat_file_sectors
                            <1>     jmp     csftdf2_read_fs_file_sectors
                            <1> %endif
                            <1>
                            <1> csftdf2_read_fat_file_sectors:
                            <1>     ; 19/03/2016
                            <1>     ; 18/03/2016
                            <1>     ; return:
                            <1>     ;   CF = 0 & EDX > 0 -> END OF FILE
                            <1>     ;   CF = 0 & EDX = 0 -> not EOF
                            <1>     ;   CF = 1 -> read error (error code in AL)
                            <1>
                            <1> csftdf2_read_fat_file_secs_0:
0000B841 8B15[58820100]     <1>     mov     edx, [csftdf_filesize]
0000B847 2B15[74820100]     <1>     sub     edx, [csftdf_sf_rbytes]
0000B84D 3B15[6C820100]     <1>     cmp     edx, [csftdf_r_size]
0000B853 7306               <1>     jnb     short csftdf2_read_fat_file_secs_1
0000B855 8915[6C820100]     <1>     mov     [csftdf_r_size], edx
                            <1>
                            <1> csftdf2_read_fat_file_secs_1:
0000B85B A1[6C820100]       <1>     mov     eax, [csftdf_r_size]
0000B860 29D2               <1>     sub     edx, edx
                            <1>     ;movzx  ecx, word [esi+LD_BPB+BytesPerSec]
                            <1>     ; 19/12/2025
0000B862 B900020000         <1>     mov     ecx, 512
0000B867 01C8               <1>     add     eax, ecx
0000B869 48                 <1>     dec     eax
0000B86A F7F1               <1>     div     ecx
0000B86C 89C1               <1>     mov     ecx, eax ; sector count
0000B86E A1[64820100]       <1>     mov     eax, [csftdf_sf_cluster]
                            <1>
                            <1>     ; EBX = memory block address (current)
                            <1>
0000B873 E8F7080000         <1>     call    read_fat_file_sectors
0000B878 7235               <1>     jc      short csftdf2_read_fat_file_secs_3
                            <1>
                            <1>     ; EBX = next memory address
                            <1>
0000B87A A1[74820100]       <1>     mov     eax, [csftdf_sf_rbytes]
0000B87F 0305[6C820100]     <1>     add     eax, [csftdf_r_size]
0000B885 8B15[58820100]     <1>     mov     edx, [csftdf_filesize]
0000B88B 39D0               <1>     cmp     eax, edx
0000B88D 7320               <1>     jnb     short csftdf2_read_fat_file_secs_3 ; edx > 0
0000B88F A3[74820100]       <1>     mov     [csftdf_sf_rbytes], eax
                            <1>
0000B894 53                 <1>     push    ebx ; *
                            <1>     ; get next cluster (csftdf_r_size! bytes)
0000B895 A1[64820100]       <1>     mov     eax, [csftdf_sf_cluster]
0000B89A E8DA060000         <1>     call    get_next_cluster
0000B89F 5B                 <1>     pop     ebx ; *
0000B8A0 7306               <1>     jnc     short csftdf2_read_fat_file_secs_2
                            <1>
                            <1>     ; 15/10/2016
                            <1>     ; Disk read error instead of drv not ready err
0000B8A2 B811000000         <1>     mov     eax, 17 ; Read error !
0000B8A7 C3                 <1>     retn
                            <1>
                            <1> csftdf2_read_fat_file_secs_2:
0000B8A8 29D2               <1>     sub     edx, edx ; 0
0000B8AA A3[64820100]       <1>     mov     [csftdf_sf_cluster], eax ; next cluster
                            <1>
                            <1> csftdf2_read_fat_file_secs_3:
0000B8AF C3                 <1>     retn
                            <1>
                            <1> csftdf2_read_sf_cluster:
                            <1>     ; 19/03/2016
0000B8B0 BB00000700         <1>     mov     ebx, Cluster_Buffer ; buffer address (64KB)
                            <1>
0000B8B5 803D[7C820100]00   <1>     cmp     byte [csftdf_percentage], 0
0000B8BC 760D               <1>     jna     short csftdf2_read_sf_clust_2
                            <1>
0000B8BE 53                 <1>     push    ebx ; *
                            <1>
                            <1> csftdf2_read_sf_clust_next:
0000B8BF E840FFFFFF         <1>     call    csftdf2_print_percentage
                            <1>
                            <1> csftdf2_read_sf_clust_0:
0000B8C4 8B35[80820100]     <1>     mov     esi, [csftdf_sf_drv_dt]
                            <1> csftdf2_read_sf_clust_1:
0000B8CA 5B                 <1>     pop     ebx ; *
                            <1>
                            <1> csftdf2_read_sf_clust_2:
0000B8CB 89DA               <1>     mov     edx, ebx
0000B8CD 0315[6C820100]     <1>     add     edx, [csftdf_r_size]
0000B8D3 81FA00000800       <1>     cmp     edx, Cluster_Buffer + 65536
0000B8D9 772B               <1>     ja      short csftdf2_write_df_cluster
                            <1>
0000B8DB E861FFFFFF         <1>     call    csftdf2_read_file_sectors ; 19/03/2016
                            <1>     ;jc     csftdf2_save_fat_file_err2 ; eocc! or disk error!
                            <1>     ; 29/07/2022
0000B8E0 723C               <1>     jc      short csftdf2_3
                            <1>
0000B8E2 09D2               <1>     or      edx, edx ; edx > 0 -> EOF
0000B8E4 7520               <1>     jnz     short csftdf2_write_df_cluster
                            <1>
0000B8E6 803D[7C820100]00   <1>     cmp     byte [csftdf_percentage], 0
0000B8ED 76DC               <1>     jna     short csftdf2_read_sf_clust_2
                            <1>
0000B8EF 53                 <1>     push    ebx ; *
                            <1>
                            <1>     ; Set cursor position
                            <1>     ; AH= 02h, BH= Page Number, DH= Row, DL= Column
0000B8F0 8A3D[7D820100]     <1>     mov     bh, [csftdf_videopage]
0000B8F6 668B15[7E820100]   <1>     mov     dx, [csftdf_cursorpos]
0000B8FD B402               <1>     mov     ah, 2
0000B8FF E8205EFFFF         <1>     call    _int10h
0000B904 EBB9               <1>     jmp     short csftdf2_read_sf_clust_next
                            <1>
                            <1> csftdf2_write_df_cluster:
                            <1>
                            <1> ; 31/08/2024
                            <1> %if 1
                            <1>     ; 27/08/2024
                            <1>     ; 19/03/2016
                            <1>     ;mov    esi, [csftdf_df_drv_dt] ; (!)
                            <1>     ;mov    ebx, Cluster_Buffer ; buffer address (64KB)
                            <1>     ;;;
                            <1>     ; 27/08/2024
0000B906 E8FF020000         <1>     call    csftdf2_update_df_fclust
                            <1>     ; 31/08/2024
0000B90B 7305               <1>     jnc     short csftdf2_update_df_fclust_ok
0000B90D E9C3020000         <1>     jmp     csftdf2_rw_error
                            <1>
```

```
                            <1> csftdf2_update_df_fclust_ok:
                            <1> %endif
                            <1>     ; esi = [csftdf_df_drv_dt]
                            <1>
                            <1>     ;mov    esi, [csftdf_df_drv_dt]
0000B912 BB00000700         <1>     mov     ebx, Cluster_Buffer ; buffer address (64KB)
                            <1>
                            <1> csftdf2_write_df_clust_next:
0000B917 E85B000000         <1>     call    csftdf2_write_file_sectors ; 19/03/2016
                            <1>     ;jc     csftdf2_save_fat_file_err2 ; eocc! or disk error!
                            <1>     ; 29/07/2022
0000B91C 7305               <1>     jnc     short csftdf2_4
                            <1> csftdf2_3:
0000B91E E939020000         <1>     jmp     csftdf2_save_fat_file_err2
                            <1> csftdf2_4:
0000B923 09D2               <1>     or      edx, edx ; edx > 0 -> EOF
0000B925 7513               <1>     jnz     short csftdf2_rw_f_clust_ok
                            <1>
0000B927 81FB00000800       <1>     cmp     ebx, Cluster_Buffer + 65536
0000B92D 72E8               <1>     jb      short csftdf2_write_df_clust_next
                            <1>
                            <1>     ; 31/08/2024
0000B92F 8B35[80820100]     <1>     mov     esi, [csftdf_sf_drv_dt]
                            <1>
0000B935 E976FFFFFF         <1>     jmp     csftdf2_read_sf_cluster ; 31/08/2024
                            <1>
                            <1> csftdf2_rw_f_clust_ok:
0000B93A 803D[7C820100]00   <1>     cmp     byte [csftdf_percentage], 0
                            <1>     ;jna    csftdf2_save_fat_file_4 ; 25/03/2016
                            <1>     ; 29/07/2022
0000B941 762A               <1>     jna     short csftdf2_5
                            <1>
                            <1>     ; "100%"
0000B943 BF[CD340100]       <1>     mov     edi, percentagestr
0000B948 B031               <1>     mov     al, '1'
0000B94A AA                 <1>     stosb
0000B94B B030               <1>     mov     al, '0'
0000B94D AA                 <1>     stosb
0000B94E AA                 <1>     stosb
                            <1>
0000B94F 8A3D[7D820100]     <1>     mov     bh, [csftdf_videopage]
0000B955 668B15[7E820100]   <1>     mov     dx, [csftdf_cursorpos]
0000B95C B402               <1>     mov     ah, 2
0000B95E E8C15DFFFF         <1>     call    _int10h
                            <1>
0000B963 BE[CD340100]       <1>     mov     esi, percentagestr
0000B968 E83DB4FFFF         <1>     call    print_msg
                            <1> csftdf2_5:
0000B96D E974010000         <1>         jmp     csftdf2_save_fat_file_4
                            <1>
                            <1> csftdf2_load_fs_file:
                            <1>     ; temporary - 18/03/2016
0000B972 E95C020000         <1>         jmp     csftdf2_read_error
                            <1>
                            <1> csftdf2_write_file_sectors:
                            <1>
                            <1> ; 19/12/2025
                            <1> %if 0
                            <1>     ; 31/08/2024
                            <1>     ; 30/08/2024
                            <1>     ; 19/03/2016
                            <1>     cmp     byte [esi+LD_FATType], 0
                            <1>     ;jna    csftdf2_write_fs_file_sectors
                            <1>     ; 29/07/2022
                            <1>     ja      short csftdf2_write_fat_file_sectors
                            <1>     jmp     csftdf2_write_fs_file_sectors
                            <1> %endif
                            <1>
                            <1> csftdf2_write_fat_file_sectors:
                            <1>     ; 30/08/2024
                            <1>     ; 19/03/2016
                            <1>     ; 18/03/2016
                            <1>     ; return:
                            <1>     ;   CF = 0 & EDX > 0 -> END OF FILE
                            <1>     ;   CF = 0 & EDX = 0 -> not EOF
                            <1>     ;   CF = 1 -> write error (error code in AL)
                            <1>
                            <1> csftdf2_write_fat_file_secs_0:
0000B977 8B15[58820100]     <1>     mov     edx, [csftdf_filesize]
0000B97D 2B15[78820100]     <1>     sub     edx, [csftdf_df_wbytes]
0000B983 3B15[70820100]     <1>     cmp     edx, [csftdf_w_size]
0000B989 7306               <1>     jnb     short csftdf2_write_fat_file_secs_1
0000B98B 8915[70820100]     <1>     mov     [csftdf_w_size], edx
                            <1>
                            <1> csftdf2_write_fat_file_secs_1:
                            <1>
                            <1> ; 31/08/2024
                            <1> %if 0
                            <1>     ;;;
                            <1>     ; 30/08/2024
                            <1>     mov     eax, [csftdf_df_cluster]
                            <1>     and     eax, eax
                            <1>     jnz     short csftdf2_write_fat_file_secs_@
                            <1>     push    ebx
                            <1>     call    add_new_cluster
                            <1>     pop     ebx
                            <1>     jc      short csftdf2_write_fat_file_secs_5
                            <1>     mov     [csftdf_df_cluster], eax
                            <1> csftdf2_write_fat_file_secs_@:
                            <1>     ;;;
                            <1> %endif
                            <1>
0000B991 A1[70820100]       <1>     mov     eax, [csftdf_w_size]
0000B996 29D2               <1>     sub     edx, edx
                            <1>     ;movzx  ecx, word [esi+LD_BPB+BytesPerSec]
                            <1>     ; 19/12/2025
0000B998 B900020000         <1>     mov     ecx, 512
0000B99D 01C8               <1>     add     eax, ecx
0000B99F 48                 <1>     dec     eax
0000B9A0 F7F1               <1>     div     ecx
0000B9A2 89C1               <1>     mov     ecx, eax ; sector count
0000B9A4 A1[68820100]       <1>     mov     eax, [csftdf_df_cluster]
                            <1>
                            <1>     ; EBX = memory block address (current)
                            <1>
0000B9A9 E8DF0E0000         <1>     call    write_fat_file_sectors
0000B9AE 7254               <1>     jc      short csftdf2_write_fat_file_secs_4
                            <1>
                            <1>     ; EBX = next memory address
                            <1>
0000B9B0 A1[78820100]       <1>     mov     eax, [csftdf_df_wbytes]
0000B9B5 0305[70820100]     <1>     add     eax, [csftdf_w_size]
0000B9BB 8B15[58820100]     <1>     mov     edx, [csftdf_filesize]
0000B9C1 39D0               <1>     cmp     eax, edx
```

```
4570 0000B9C3 733F                <1>     jnb     short csftdf2_write_fat_file_secs_4
4571 0000B9C5 A3[78820100]        <1>     mov     [csftdf_df_wbytes], eax
4572                              <1>     ;
4573 0000B9CA A3[1A820100]        <1>     mov     [DestinationFile_DirEntry+DirEntry_FileSize], eax
4574                              <1>
4575                              <1>
4576 0000B9CF 53                  <1>     push    ebx ; *
4577                              <1>
4578                              <1>     ; 30/08/2024
4579 0000B9D0 A1[68820100]        <1>     mov     eax, [csftdf_df_cluster] ; last cluster
4580                              <1>
4581 0000B9D5 803D[56820100]01    <1>     cmp     byte [DestinationFileFound], 1
4582 0000B9DC 720B                <1>     jb      short csftdf2_write_fat_file_secs_2
4583                              <1>
4584                              <1>     ; get next cluster (csftdf_w_size! bytes)
4585                              <1>     ;mov    eax, [csftdf_df_cluster]
4586 0000B9DE E896050000          <1>     call    get_next_cluster
4587 0000B9E3 7317                <1>     jnc     short csftdf2_write_fat_file_secs_3
4588                              <1>
4589 0000B9E5 21C0                <1>     and     eax, eax ; end of cluster chain!?
4590 0000B9E7 751C                <1>     jnz     short csftdf2_write_fat_file_secs_5 ; disk error !
4591                              <1>
4592                              <1> csftdf2_write_fat_file_secs_2:
4593                              <1>     ;mov    eax, [csftdf_df_cluster] ; last cluster
4594                              <1>     ; 30/08/2024
4595 0000B9E9 E80E0E0000          <1>     call    add_new_cluster
4596 0000B9EE 7215                <1>     jc      short csftdf2_write_fat_file_secs_5
4597                              <1>
4598                              <1>     ; NOTE: Destination file size may be bigger than
4599                              <1>     ; source file size when the last reading fails after here.
4600                              <1>     ; (The last -empty- cluster of destination file must be
4601                              <1>     ; truncated and LMDT must be current date&time for partial
4602                              <1>     ; copy result!)
4603 0000B9F0 8B15[70820100]      <1>     mov     edx, [csftdf_w_size] ; < = bytes per cluster
4604 0000B9F6 0115[1A820100]      <1>     add     [DestinationFile_DirEntry+DirEntry_FileSize], edx
4605                              <1>
4606                              <1> csftdf2_write_fat_file_secs_3:
4607                              <1>     ; 30/08/2024
4608                              <1>     ;pop    ebx ; *
4609 0000B9FC 29D2                <1>     sub     edx, edx ; 0
4610 0000B9FE A3[68820100]        <1>     mov     [csftdf_df_cluster], eax ; next cluster
4611                              <1>
4612                              <1>     ; 30/08/2024
4613 0000BA03 5B                  <1>     pop     ebx ; *
4614                              <1>
4615                              <1> csftdf2_write_fat_file_secs_4:
4616 0000BA04 C3                  <1>     retn
4617                              <1>
4618                              <1> csftdf2_write_fat_file_secs_5:
4619 0000BA05 5B                  <1>     pop     ebx ; *
4620 0000BA06 B812000000          <1>     mov     eax, 18 ; Write error !
4621                              <1>     ; 16/10/2016 (1Dh -> 18)
4622 0000BA0B C3                  <1>     retn
4623                              <1>
4624                              <1> csftdf2_save_file:
4625                              <1>     ; 19/12/2025
4626                              <1>     ; 31/08/2024
4627                              <1>     ; 26/08/2024, 30/08/2024
4628                              <1>     ; 09/12/2017
4629                              <1>     ; 19/03/2016, 25/03/2016
4630                              <1>     ; 18/03/2016
4631 0000BA0C 8B35[84820100]      <1>     mov     esi, [csftdf_df_drv_dt] ; logical dos drv desc. tbl.
4632 0000BA12 8B1D[5C820100]      <1>     mov     ebx, [csftdf_sf_mem_addr] ; memory block address
4633                              <1>
4634                              <1> ; 19/12/2025
4635                              <1> %if 0
4636                              <1>     cmp     byte [esi+LD_FATType], 0
4637                              <1>     ;jna    csftdf2_save_fs_file
4638                              <1>     ; 29/07/2022
4639                              <1>     ja      short csftdf2_save_fat_file
4640                              <1>     jmp     csftdf2_save_fs_file
4641                              <1> %endif
4642                              <1>
4643                              <1> csftdf2_save_fat_file:
4644                              <1>
4645                              <1> ; 31/08/2024
4646                              <1> %if 1
4647                              <1>     ; 30/08/2024
4648 0000BA18 53                  <1>     push    ebx ; *
4649                              <1>     ;;;
4650                              <1>     ; 27/08/2024
4651                              <1>     ;call   csftdf2_update_df_fclust
4652 0000BA19 E8F2010000          <1>     call    csftdf2_update_df_fclust_@ ; 31/08/2024
4653 0000BA1E 5B                  <1>     pop     ebx ; *
4654 0000BA1F 7236                <1>     jc      short csftdf2_save_fat_file_err
4655                              <1>     ; esi = [csftdf_df_drv_dt] ; (*)
4656                              <1>     ;;;
4657                              <1> %endif
4658                              <1>
4659 0000BA21 803D[7C820100]00    <1>     cmp     byte [csftdf_percentage], 0
4660 0000BA28 7726                <1>     ja      short csftdf2_save_fat_file_0 ; 30/08/2024
4661                              <1>
4662 0000BA2A 53                  <1>     push    ebx ; **
4663                              <1>
4664                              <1>     ; Set cursor position
4665                              <1>     ; AH= 02h, BH= Page Number, DH= Row, DL= Column
4666 0000BA2B 8A3D[7D820100]      <1>     mov     bh, [csftdf_videopage]
4667 0000BA31 668B15[7E820100]    <1>     mov     dx, [csftdf_cursorpos]
4668 0000BA38 B402                <1>     mov     ah, 2
4669 0000BA3A E8E55CFFFF          <1>     call    _int10h
4670                              <1>
4671 0000BA3F BE[C1340100]        <1>     mov     esi, msg_writing
4672 0000BA44 E861B3FFFF          <1>     call    print_msg
4673                              <1>
4674                              <1> csftdf2_save_fat_file_next:
4675 0000BA49 5B                  <1>     pop     ebx ; **
4676                              <1>
4677                              <1>     ; 31/08/2024
4678 0000BA4A 8B35[84820100]      <1>     mov     esi, [csftdf_df_drv_dt]
4679                              <1>
4680                              <1> csftdf2_save_fat_file_0: ; 30/08/2024
4681                              <1>
4682                              <1> csftdf2_save_fat_file_1:
4683 0000BA50 E822FFFFFF          <1>     call    csftdf2_write_file_sectors ; 19/03/2016
4684                              <1>     ;jc     csftdf2_rw_error ; eocc! or disk error!
4685                              <1>     ; 29/07/2022
4686 0000BA55 7305                <1>     jnc     short csftdf2_6
4687                              <1> csftdf2_save_fat_file_err: ; 27/08/2024
4688 0000BA57 E979010000          <1>     jmp     csftdf2_rw_error
4689                              <1>
4690                              <1> csftdf2_6:
4691 0000BA5C 09D2                <1>     or      edx, edx ; edx > 0 -> EOF
4692 0000BA5E 756C                <1>         jnz     short csftdf2_save_fat_file_3 ; 25/03/2016
4693                              <1>
```

```
4694 0000BA60 803D[7C820100]00    <1>     cmp     byte [csftdf_percentage], 0
4695 0000BA67 76E7                <1>     jna     short csftdf2_save_fat_file_1
4696                              <1>
4697 0000BA69 B020                <1>     mov     al, 20h
4698 0000BA6B BF[CD340100]        <1>     mov     edi, percentagestr
4699 0000BA70 AA                  <1>     stosb
4700 0000BA71 AA                  <1>     stosb
4701 0000BA72 A1[78820100]        <1>     mov     eax, [csftdf_df_wbytes]
4702                              <1>     ;mov    edx, 100
4703                              <1>     ; 29/07/2022
4704 0000BA77 31D2                <1>     xor     edx, edx
4705 0000BA79 B264                <1>     mov     dl, 100
4706 0000BA7B F7E2                <1>     mul     edx
4707 0000BA7D 8B0D[58820100]      <1>     mov     ecx, [csftdf_filesize]
4708 0000BA83 F7F1                <1>     div     ecx
4709 0000BA85 B10A                <1>     mov     cl, 10
4710 0000BA87 F6F1                <1>     div     cl
4711 0000BA89 80C430              <1>     add     ah, '0'
4712 0000BA8C 8827                <1>     mov     [edi], ah
4713 0000BA8E 20C0                <1>     and     al, al
4714 0000BA90 740A                <1>     jz      short csftdf2_save_fat_file_2
4715 0000BA92 4F                  <1>     dec     edi
4716                              <1>     ;cbw
4717 0000BA93 30E4                <1>     xor     ah, ah ; 09/12/2017
4718 0000BA95 F6F1                <1>     div     cl
4719 0000BA97 80C430              <1>     add     ah, '0'
4720 0000BA9A 8827                <1>     mov     [edi], ah
4721                              <1>     ;and    al, al
4722                              <1>     ;jz     short csftdf2_save_fat_file_2
4723                              <1>     ;dec    edi
4724                              <1>     ;mov    [edi], '1' ; 100%
4725                              <1>
4726                              <1> csftdf2_save_fat_file_2:
4727 0000BA9C 53                  <1>     push    ebx ; *
4728                              <1>
4729 0000BA9D E802000000          <1>     call    csftdf2_print_wr_percentage ; 25/03/2016
4730                              <1>
4731 0000BAA2 EBA5                <1>         jmp     csftdf2_save_fat_file_next
4732                              <1>
4733                              <1> csftdf2_print_wr_percentage:
4734                              <1>     ; Set cursor position
4735                              <1>     ; AH= 02h, BH= Page Number, DH= Row, DL= Column
4736 0000BAA4 8A3D[7D820100]      <1>     mov     bh, [csftdf_videopage]
4737 0000BAAA 668B15[7E820100]    <1>     mov     dx, [csftdf_cursorpos]
4738 0000BAB1 B402                <1>     mov     ah, 2
4739 0000BAB3 E86C5CFFFF          <1>     call    _int10h
4740                              <1>
4741 0000BAB8 BE[C1340100]        <1>     mov     esi, msg_writing
4742 0000BABD E8E8B2FFFF          <1>     call    print_msg
4743                              <1>
4744 0000BAC2 BE[CD340100]        <1>     mov     esi, percentagestr
4745                              <1>     ;call   print_msg
4746                              <1>     ;retn
4747 0000BAC7 E9DEB2FFFF          <1>     jmp     print_msg
4748                              <1>
4749                              <1> csftdf2_save_fat_file_3:
4750 0000BACC 803D[7C820100]00    <1>     cmp     byte [csftdf_percentage], 0
4751                              <1>         ;jna       csftdf2_save_fat_file_4 ; 25/03/2016
4752                              <1>     ; 29/07/2022
4753                              <1>     ;ja     short csftdf2_7
4754                              <1>     ;jmp    csftdf2_save_fat_file_4
4755                              <1>     ; 31/08/2024
4756 0000BAD3 7611                <1>     jna     short csftdf2_save_fat_file_4
4757                              <1>
4758                              <1> csftdf2_7:
4759                              <1>     ; "100%"
4760 0000BAD5 BF[CD340100]        <1>     mov     edi, percentagestr
4761 0000BADA B031                <1>     mov     al, '1'
4762 0000BADC AA                  <1>     stosb
4763 0000BADD B030                <1>     mov     al, '0'
4764 0000BADF AA                  <1>     stosb
4765 0000BAE0 AA                  <1>     stosb
4766                              <1>
4767 0000BAE1 E8BEFFFFFF          <1>     call    csftdf2_print_wr_percentage
4768                              <1>
4769                              <1> csftdf2_save_fat_file_4:
4770 0000BAE6 803D[56820100]00    <1>     cmp     byte [DestinationFileFound], 0
4771 0000BAED 7645                <1>     jna     short csftdf2_save_fat_file_6
4772                              <1>
4773 0000BAEF 8B35[84820100]      <1>     mov     esi, [csftdf_df_drv_dt] ; 31/03/2016
4774                              <1>
4775 0000BAF5 A1[68820100]        <1>     mov     eax, [csftdf_df_cluster] ; last cluster
4776 0000BAFA E87A040000          <1>     call    get_next_cluster
4777 0000BAFF 7233                <1>     jc      short csftdf2_save_fat_file_6 ; eocc! or disk error!
4778                              <1>
4779 0000BB01 A1[68820100]        <1>     mov     eax, [csftdf_df_cluster] ; last cluster
4780                              <1>     ; 29/08/2024
4781 0000BB06 31C9                <1>     xor     ecx, ecx
4782                              <1>     ;mov    [FAT_ClusterCounter], ecx ; 0 ; reset
4783 0000BB08 49                  <1>     dec     ecx ; 0FFFFFFFFh
4784                              <1>     ;shr    ecx, 4 ; 28 bit ; 0FFFFFFFh
4785                              <1>     ; 29/08/2024
4786                              <1>     ; (update_cluster will save 28 bit cluster value)
4787                              <1>     ;mov    ecx, 0FFFFFFFh
4788 0000BB09 E852070000          <1>     call    update_cluster
4789 0000BB0E 7224                <1>     jc      short csftdf2_save_fat_file_6 ; really last cluster!?
4790                              <1>
4791 0000BB10 A3[68820100]        <1>     mov     [csftdf_df_cluster], eax ; next cluster
4792                              <1>
4793                              <1>     ; byte [FAT_BuffValidData] = 2
4794 0000BB15 E8BA090000          <1>     call    save_fat_buffer
4795 0000BB1A 730E                <1>     jnc     short csftdf2_save_fat_file_5
4796                              <1>
4797 0000BB1C 8B15[58820100]      <1>     mov     edx, [csftdf_filesize]
4798 0000BB22 8915[1A820100]      <1>     mov     [DestinationFile_DirEntry+DirEntry_FileSize], edx
4799 0000BB28 EB57                <1>     jmp     short csftdf2_save_fat_file_err3
4800                              <1>
4801                              <1> csftdf2_save_fat_file_5:
4802 0000BB2A A1[68820100]        <1>     mov     eax, [csftdf_df_cluster]
4803                              <1>
4804                              <1>     ; EAX = First cluster to be truncated/unlinked
4805                              <1>     ; ESI = Logical dos drive description table address
4806 0000BB2F E8DA0B0000          <1>     call    truncate_cluster_chain
4807                              <1>
4808                              <1> csftdf2_save_fat_file_6:
4809                              <1>     ; 28/03/2016
4810 0000BB34 BE[89810100]        <1>     mov     esi, SourceFile_DirEntry+DirEntry_Attr ; +11 to + 18
4811 0000BB39 BF[09820100]        <1>     mov     edi, DestinationFile_DirEntry+DirEntry_Attr ; +11 to + 18
4812 0000BB3E A4                  <1>     movsb ; +11
4813 0000BB3F A5                  <1>     movsd ; +12 .. +15
4814 0000BB40 66A5                <1>     movsw ; +16 .. +17
4815                              <1>           ; + 18
4816 0000BB42 83C604              <1>     add     esi, 4
4817 0000BB45 83C704              <1>     add     edi, 4
```

```
4818 0000BB48 A5                  <1>     movsd   ; DirEntry_WrtTime ; +22 .. +25
4819                              <1>
4820 0000BB49 8B15[58820100]      <1>     mov     edx, [csftdf_filesize]
4821 0000BB4F 8915[1A820100]      <1>     mov     [DestinationFile_DirEntry+DirEntry_FileSize], edx
4822                              <1>
4823 0000BB55 E892F1FFFF          <1>     call    convert_current_date_time
4824                              <1>     ; DX = Date in dos dir entry format
4825                              <1>     ; AX = Time in dos dir entry format
4826 0000BB5A EB4C                <1>     jmp     short csftdf2_save_fat_file_7
4827                              <1>
4828                              <1>     ; 31/08/2024
4829                              <1> ;csftdf2_save_fat_file_err1:
4830                              <1>     ;pop    ebx ; *
4831                              <1>
4832                              <1> csftdf2_save_fat_file_err2:
4833 0000BB5C A1[78820100]        <1>     mov     eax, [csftdf_df_wbytes]
4834 0000BB61 8B15[1A820100]      <1>     mov     edx, [DestinationFile_DirEntry+DirEntry_FileSize]
4835 0000BB67 39C2                <1>     cmp     edx, eax
4836 0000BB69 7616                <1>     jna     short csftdf2_save_fat_file_err3
4837 0000BB6B A1[68820100]        <1>     mov     eax, [csftdf_df_cluster] ; last (empty) cluster
4838                              <1>     ; ESI = Logical dos drive description table address
4839 0000BB70 E8990B0000          <1>     call    truncate_cluster_chain
4840 0000BB75 720A                <1>     jc      short csftdf2_save_fat_file_err3
4841 0000BB77 A1[78820100]        <1>     mov     eax, [csftdf_df_wbytes]
4842 0000BB7C A3[1A820100]        <1>     mov     [DestinationFile_DirEntry+DirEntry_FileSize], eax
4843                              <1> csftdf2_save_fat_file_err3:
4844 0000BB81 E866F1FFFF          <1>     call    convert_current_date_time
4845                              <1>     ; DX = Date in dos dir entry format
4846                              <1>     ; AX = Time in dos dir entry format
4847 0000BB86 C605[0B820100]00    <1>     mov     byte [DestinationFile_DirEntry+DirEntry_CrtTimeTenth], 0
4848 0000BB8D 66A3[0C820100]      <1>     mov     [DestinationFile_DirEntry+DirEntry_CrtTime], ax
4849 0000BB93 668915[0E820100]    <1>     mov     [DestinationFile_DirEntry+DirEntry_CrtDate], dx
4850 0000BB9A 66A3[14820100]      <1>     mov     [DestinationFile_DirEntry+DirEntry_WrtTime], ax
4851 0000BBA0 668915[16820100]    <1>     mov     [DestinationFile_DirEntry+DirEntry_WrtDate], dx
4852 0000BBA7 F9                  <1>     stc
4853                              <1> csftdf2_save_fat_file_7:
4854 0000BBA8 9C                  <1>     pushf
4855 0000BBA9 668915[10820100]    <1>     mov     [DestinationFile_DirEntry+DirEntry_LastAccDate], dx
4856 0000BBB0 BE[FE810100]        <1>     mov     esi, DestinationFile_DirEntry
4857 0000BBB5 BF00000800          <1>     mov     edi, Directory_Buffer
4858 0000BBBA 0FB70D[26820100]    <1>     movzx   ecx, word [DestinationFile_DirEntryNumber] ; (<2048)
4859                              <1>     ;shl    cx, 5 ; 32 * directory entry number
4860                              <1>     ; 29/07/2022
4861 0000BBC1 C1E105              <1>     shl     ecx, 5
4862 0000BBC4 01CF                <1>     add     edi, ecx
4863                              <1>     ;mov    ecx, 8
4864                              <1>     ;mov    cx, 8
4865                              <1>     ; 31/08/2024
4866                              <1>     ; 29/07/2022
4867 0000BBC6 29C9                <1>     sub     ecx, ecx
4868                              <1>     ;sub    ch, ch
4869 0000BBC8 B108                <1>     mov     cl, 8
4870 0000BBCA F3A5                <1>     rep     movsd
4871 0000BBCC 9D                  <1>     popf
4872 0000BBCD 730B                <1>     jnc     short csftdf2_write_file_OK
4873                              <1>
4874                              <1> csftdf2_write_error:
4875                              <1>     ; 18/03/2016
4876                              <1>     ;mov    al, 1Dh ; write error
4877                              <1>     ; 31/08/2024
4878 0000BBCF B012                <1>     mov     al, ERR_DRV_WRITE ; 18 ; write error
4879 0000BBD1 EB02                <1>     jmp     short csftdf2_rw_error
4880                              <1>
4881                              <1>     ; 16/03/2016
4882                              <1> csftdf2_read_error:
4883 0000BBD3 B011                <1>     mov     al, 17 ; ; Drive not ready or read error!
4884                              <1> csftdf2_rw_error:
4885 0000BBD5 A2[55820100]        <1>     mov     [csftdf_rw_err], al
4886                              <1>
4887                              <1> csftdf2_write_file_OK:
4888                              <1>     ; 18/03/2016
4889 0000BBDA C605[697E0100]02    <1>     mov     byte [DirBuff_ValidData], 2
4890 0000BBE1 E89BF1FFFF          <1>     call    save_directory_buffer
4891                              <1>
4892                              <1>     ; Update last modification date&time of destination
4893                              <1>     ; file's (parent) directory
4894 0000BBE6 E81FF2FFFF          <1>     call    update_parent_dir_lmdt
4895                              <1>     ;
4896 0000BBEB A1[5C820100]        <1>     mov     eax, [csftdf_sf_mem_addr] ; start address
4897                              <1>
4898 0000BBF0 21C0                <1>     and     eax, eax
4899 0000BBF2 7509                <1>     jnz     short csftdf2_dealloc_mblock
4900                              <1>
4901                              <1>     ; 31/08/2024
4902                              <1>     ;mov    ch, al ; 0 (Cluster r/w, not full loading)
4903                              <1> csftdf2_dealloc_retn:
4904                              <1>     ;mov    cl, [csftdf_rw_err]
4905                              <1>     ;mov    eax, [csftdf_df_cluster]
4906                              <1>     ; 31/08/2024
4907 0000BBF4 A0[55820100]        <1>     mov     al, [csftdf_rw_err]
4908 0000BBF9 3C01                <1>     cmp     al, 1
4909 0000BBFB F5                  <1>     cmc
4910                              <1>     ; if al > 0 -> cf = 1
4911                              <1> csftdf2_write_df_error:   ; 31/08/2024
4912                              <1> csftdf2_write_df_clust_@@: ; 31/08/2024
4913 0000BBFC C3                  <1>     retn
4914                              <1>
4915                              <1> csftdf2_dealloc_mblock:
4916 0000BBFD 8B0D[60820100]      <1>     mov     ecx, [csftdf_sf_mem_bsize] ; block size
4917 0000BC03 E84BA3FFFF          <1>     call    deallocate_memory_block
4918                              <1>     ; 31/08/2024
4919                              <1>         ;mov       ch, 0FFh ; (File was full loaded at memory)
4920 0000BC08 EBEA                <1>     jmp     short csftdf2_dealloc_retn
4921                              <1>
4922                              <1>     ;;;;
4923                              <1> csftdf2_update_df_fclust:
4924                              <1>     ; 31/08/2024
4925                              <1>     ; 27/08/2024 - TRDOS 386 v2.0.9
4926                              <1>     ; update (directory entry of)
4927                              <1>     ; the new (created) destination file
4928                              <1>     ; with new first cluster (it was 0 before)
4929                              <1>
4930 0000BC0A 8B35[84820100]      <1>     mov     esi, [csftdf_df_drv_dt]       ; (!)
4931                              <1>
4932                              <1> csftdf2_update_df_fclust_@: ; 31/08/2024
4933 0000BC10 A1[68820100]        <1>     mov     eax, [csftdf_df_cluster]
4934 0000BC15 21C0                <1>     and     eax, eax
4935 0000BC17 75E3                <1>     jnz     short csftdf2_write_df_clust_@@
4936                              <1>     ; if eax = 0, add new cluster
4937                              <1> ; 31/08/2024
4938                              <1> %if 0
4939                              <1>     call    get_first_free_cluster
4940                              <1>     jc      short csftdf2_write_df_error
4941                              <1>     ; EAX >= 2 and EAX < FFFFFFFFh is valid
```

```
                                <1>
                                <1>     cmp     eax, 0FFFFFFFFh ; no free space
                                <1>     jne     short csftdf2_write_df_clust_@
                                <1>     inc     eax ; 0
                                <1>     stc
                                <1>     mov     al, 27h ; insufficient disk space
                                <1> csftdf2_write_df_error:
                                <1>     retn
                                <1> %else
                                <1>     ; 31/08/2024
0000BC19 E8DE0B0000             <1>     call    add_new_cluster
0000BC1E 72DC                   <1>     jc      short csftdf2_write_df_error
                                <1> %endif
                                <1>
                                <1> csftdf2_write_df_clust_@:
                                <1>     ; 31/08/2024
0000BC20 A3[68820100]           <1>     mov     [csftdf_df_cluster], eax
                                <1>     ;
                                <1>     ; eax = (new) first cluster
0000BC25 56                     <1>     push    esi
0000BC26 BE[FE810100]           <1>     mov     esi, DestinationFile_DirEntry
0000BC2B 0FB73D[26820100]       <1>     movzx   edi, word [DestinationFile_DirEntryNumber]
0000BC32 C1E705                 <1>     shl     edi, 5 ; 32 * index number
0000BC35 81C700000800           <1>     add     edi, Directory_Buffer
0000BC3B C1C010                 <1>     rol     eax, 16
0000BC3E 66894614               <1>     mov     [esi+DirEntry_FstClusHI],ax
0000BC42 C1C810                 <1>     ror     eax, 16
0000BC45 6689461A               <1>     mov     [esi+DirEntry_FstClusLO],ax
0000BC49 B908000000             <1>     mov     ecx, 8 ; 32 bytes
0000BC4E F3A5                   <1>     rep     movsd
0000BC50 5E                     <1>     pop     esi
                                <1>     ;
0000BC51 C605[697E0100]02       <1>     mov     byte [DirBuff_ValidData], 2 ; change sign
                                <1>
                                <1> ; 31/08/2024
                                <1> %if 0
                                <1>     call    save_directory_buffer
                                <1>     jc      short csftdf2_write_df_error
                                <1>     ;
                                <1>     ; +1 cluster is in use (as first cluster)
                                <1>     mov     ebx, [esi+LD_FreeSectors]
                                <1>     inc     ebx  ; 0FFFFFFFFh ? invalid ?
                                <1>     jz      short csftdf2_write_df_clust_@@ ; yes
                                <1>     xor     ebx, ebx
                                <1>     mov     bl, [esi+LD_BPB+BPB_SecPerClust]
                                <1>     sub     [esi+LD_FreeSectors], ebx
                                <1>     ;
                                <1> csftdf2_write_df_clust_@@:
                                <1>     retn
                                <1>     ;;;;
                                <1> %else
                                <1>     ; 31/08/2024
0000BC58 E924F1FFFF             <1>     jmp     save_directory_buffer
                                <1> %endif
                                <1>
                                <1> csftdf2_save_fs_file:
                                <1>     ; 16/10/2016 (1Dh -> 18)
                                <1>     ; temporary - (21/03/2016)
                                <1>     ;mov    eax, 18 ; write error
                                <1>     ; 29/07/2022
0000BC5D 31C0                   <1>     xor     eax, eax
0000BC5F B012                   <1>     mov     al, 18
0000BC61 F9                     <1>     stc
0000BC62 C3                     <1>     retn
                                <1>
                                <1> create_file:
                                <1>     ; 19/12/2025 (TRDOS 386 v2.0.10)
                                <1>     ; 31/08/2024
                                <1>     ; 29/08/2024
                                <1>     ; 27/08/2024
                                <1>     ; 26/08/2024 (TRDOS 386 v2.0.9)
                                <1>     ; 30/07/2022
                                <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <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>     ;       EAX = New file's first cluster
                                <1>     ;               (0 for empty file) ; 29/08/2024
                                <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>     ;       ; 29/08/2024
                                <1>     ;       EBX = File Size (0 for a new, empty file)
                                <1>     ;       ECX = Directory Entry Index/Number (<2048)
                                <1>     ;            (in directory cluster, not in directory)
                                <1>     ;       EDX = Directory Cluster Number (of the file)
                                <1>     ;
                                <1>     ;       cf = 1 -> error code in AL (EAX)
                                <1>     ;
                                <1>     ; (Modified registers: eax, ebx, ecx, edx, esi, edi)
                                <1>     ;
                                <1>
                                <1> ;   test    cl, 18h (directory or volume name)
                                <1> ;   jnz     short loc_createfile_access_denied
0000BC63 80E107                 <1>     and     cl, 07h ; S, H, R
0000BC66 880D[A4820100]         <1>         mov     [createfile_attrib], cl
                                <1>
0000BC6C 89D9                   <1>     mov     ecx, ebx
0000BC6E 89F3                   <1>     mov     ebx, esi ; ASCIIZ File Name address
0000BC70 29D2                   <1>     sub     edx, edx
0000BC72 8A35[42770100]         <1>         mov     dh, [Current_Drv]
0000BC78 BE00010900             <1>         mov     esi, Logical_DOSDisks
0000BC7D 01D6                   <1>     add     esi, edx
                                <1>
0000BC7F 8815[AD820100]         <1>     mov     [createfile_UpdatePDir], dl ; 0 ; 31/03/2016
                                <1>
                                <1>     ; LD_DiskType = 0 for write protection (read only)
0000BC85 807E0101               <1>     cmp     byte [esi+LD_DiskType], 1 ; 0 = Invalid
0000BC89 7308                   <1>     jnb     short loc_createfile_check_file_sytem
                                <1>     ; 16/10/2016 (TRDOS Error code: 30, disk write protected)
                                <1>     ;mov    eax, 30 ; 13h, MSDOS err : Disk write-protected
                                <1>     ;mov    dx, 0
```

```
                                <1>     ; 29/07/2022
0000BC8B 31D2                   <1>     xor     edx, edx ; 0
0000BC8D 31C0                   <1>     xor     eax, eax ; 0
0000BC8F B01E                   <1>     mov     al, 30
0000BC91 F9                     <1>     stc
                                <1>     ; err retn: EDX = 0, EBX = File name offset
                                <1>     ; ESI -> Dos drive description table address
0000BC92 C3                     <1>     retn
                                <1>
                                <1> ;loc_createfile_access_denied:
                                <1> ;   mov     eax, 05h ; access denied (invalid attributes input)
                                <1> ;   stc
                                <1> ;   retn
                                <1>
                                <1> loc_createfile_check_file_sytem:
                                <1>     ; 27/08/2024
0000BC93 A3[90820100]           <1>     mov     [createfile_size], eax
                                <1>
                                <1> ; 19/12/2025
                                <1> %if 0
                                <1>     cmp     byte [esi+LD_FATType], 1
                                <1>     jnb     short loc_createfile_chk_empty_FAT_file_sign1
                                <1>
                                <1>     ; 27/08/2024
                                <1>     ;mov    [createfile_size], eax
                                <1>     ; ESI = Logical Dos Drive Description Table address
                                <1>     ; EBX = ASCIIZ File Name address
                                <1>     jmp     create_fs_file
                                <1> %endif
                                <1>
                                <1> loc_createfile_chk_empty_FAT_file_sign1:
                                <1>     ; ECX = FFFFFFFFh -> create empty file if drive has FAT fs
0000BC98 41                     <1>     inc     ecx
0000BC99 7506                   <1>     jnz     short loc_createfile_chk_empty_FAT_file_sign2
                                <1>
0000BC9B 890D[90820100]         <1>     mov     [createfile_size], ecx ; 0  ; empty file
                                <1>
                                <1> loc_createfile_chk_empty_FAT_file_sign2:
                                <1>     ; 19/12/2025
                                <1>     ; 23/03/2016
                                <1>     ;mov    cx, [esi+LD_BPB+BytesPerSec]
                                <1>     ;mov    [createfile_BytesPerSec], cx
                                <1>
                                <1>     ; EBX = ASCIIZ File Name address
0000BCA1 0FB65613               <1>     movzx   edx, byte [esi+LD_BPB+SecPerClust]
0000BCA5 8815[A5820100]         <1>     mov     [createfile_SecPerClust], dl
0000BCAB 8B4E74                 <1>     mov     ecx, [esi+LD_FreeSectors]
0000BCAE 39D1                   <1>     cmp     ecx, edx ; byte [createfile_SecPerClust]
0000BCB0 7306                   <1>     jnb     short loc_create_fat_file
                                <1>
                                <1> loc_createfile_insufficient_disk_space:
0000BCB2 B827000000             <1>     mov     eax, 27h
                                <1> loc_createfile_gffc_retn:
0000BCB7 C3                     <1>     retn
                                <1>
                                <1> loc_create_fat_file:
0000BCB8 891D[88820100]         <1>     mov     [createfile_Name_Offset], ebx
0000BCBE 890D[8C820100]         <1>     mov     [createfile_FreeSectors], ecx
                                <1>
                                <1>     ; 27/08/2024
                                <1> ;loc_createfile_gffc_1:
                                <1>     ;call   get_first_free_cluster
                                <1>     ;jc     short loc_createfile_gffc_retn
                                <1>     ;
                                <1>     ;mov    [createfile_FFCluster], eax
                                <1>
                                <1> loc_createfile_locate_ffe_on_directory:
                                <1>     ; Current directory fcluster <> Directory buffer cluster
                                <1>     ; Current directory will be reloaded by
                                <1>     ; 'locate_current_dir_file' procedure
                                <1>     ;
                                <1>     ; ESI = Logical Dos Drv Desc. Table Adress
0000BCC4 56                     <1>     push    esi ; *
0000BCC5 31C0                   <1>     xor     eax, eax
                                <1>
0000BCC7 A3[5E7E0100]           <1>     mov     [FAT_ClusterCounter], eax ; 0
                                <1>     ; 21/03/2016
0000BCCC A2[AC820100]           <1>     mov     [createfile_wfc], al ; 0
                                <1>
0000BCD1 89C1                   <1>     mov     ecx, eax
0000BCD3 6649                   <1>     dec     cx ; FFFFh
                                <1>     ; CX = FFFFh -> find first deleted or free entry
                                <1>     ; ESI would be ASCIIZ filename address if the call
                                <1>     ; would not be for first free or deleted dir entry
0000BCD5 E81EE9FFFF             <1>     call    locate_current_dir_file
                                <1>     ;jnc    loc_createfile_set_ff_dir_entry
                                <1>     ; 29/07/2022
0000BCDA 7205                   <1>     jc      short loc_createfile_locate_file_err
                                <1>     ; 26/08/2024
                                <1>     ; cl = 0FFh, ch=0 or ch=0E5h
                                <1>     ; 27/08/2024
                                <1>     ;xor    ecx, ecx ; 26/08/2024
0000BCDC E9DA000000             <1>     jmp     loc_createfile_set_ff_dir_entry
                                <1>
                                <1> loc_createfile_locate_file_err: ; 29/07/2022
0000BCE1 5E                     <1>     pop     esi ; *
                                <1>     ; ESI = Logical DOS Drv. Description Table Address
0000BCE2 83F802                 <1>     cmp     eax, 2
0000BCE5 7402                   <1>     je      short loc_createfile_add_new_cluster
                                <1> loc_createfile_locate_file_stc_retn:
0000BCE7 F9                     <1>     stc
0000BCE8 C3                     <1>     retn
                                <1>
                                <1> loc_createfile_add_new_cluster:
0000BCE9 803D[41770100]02       <1>     cmp     byte [Current_FATType], 2
                                <1>     ;cmp    byte [esi+LD_FATType], 2
0000BCF0 770C                   <1>     ja      short loc_createfile_add_new_cluster_check_fsc
0000BCF2 803D[40770100]01       <1>     cmp     byte [Current_Dir_Level], 1
                                <1>     ;cmp    byte [esi+LD_CDirLevel], 1
0000BCF9 7303                   <1>     jnb     short loc_createfile_add_new_cluster_check_fsc
                                <1>
                                <1>     ;mov    eax, 12
0000BCFB B00C                   <1>     mov     al, 12 ; No more files
                                <1>
                                <1> loc_createfile_anc_retn:
0000BCFD C3                     <1>     retn
                                <1>
                                <1> loc_createfile_add_new_cluster_check_fsc:
0000BCFE 8B0D[8C820100]         <1>     mov     ecx, [createfile_FreeSectors]
0000BD04 0FB605[A5820100]       <1>     movzx   eax, byte [createfile_SecPerClust]
                                <1>     ;shl    ax, 1 ; AX = 2 * AX
                                <1>     ; 29/07/2022
0000BD0B D1E0                   <1>     shl     eax, 1
0000BD0D 39C1                   <1>     cmp     ecx, eax
```

```
5190 0000BD0F 72A1                <1>         jb  short loc_createfile_insufficient_disk_space
5191                              <1>
5192                              <1> loc_createfile_add_new_subdir_cluster:
5193 0000BD11 8B15[6E7E0100]      <1>     mov     edx, [DirBuff_Cluster]
5194 0000BD17 8915[98820100]      <1>     mov     [createfile_LastDirCluster], edx
5195                              <1>
5196                              <1>     ; 27/08/2024
5197                              <1>     ;;;
5198 0000BD1D E869040000          <1>     call    get_first_free_cluster
5199 0000BD22 72D9                <1>     jc      short loc_createfile_anc_retn
5200 0000BD24 A3[94820100]        <1>     mov     [createfile_FFCluster], eax
5201                              <1>     ;;;
5202                              <1>     ; 27/08/2024
5203                              <1>     ;;;
5204                              <1>     ;mov    eax, [createfile_FFCluster]
5205                              <1>     ; eax = cluster address of the new directory sector
5206                              <1>     ;;;
5207                              <1>
5208 0000BD29 E807040000          <1>     call    load_FAT_sub_directory
5209 0000BD2E 72CD                <1>     jc      short loc_createfile_anc_retn
5210                              <1>
5211                              <1>     ; 27/08/2024
5212                              <1>     ; [DirBuff_Cluster] = [createfile_FFCluster]
5213                              <1>
5214                              <1> pass_createfile_add_new_subdir_cluster:
5215                              <1>     ; clear directory buffer (new dir sector) ; 27/08/2024
5216                              <1>     ;;movzx eax, word [esi+LD_BPB+BytesPerSec]
5217                              <1>     ;movzx  eax, word [createfile_BytesPerSec] ; 23/03/2016
5218                              <1>     ; 19/12/2025
5219                              <1>     ;mov    eax, 512
5220                              <1>     ; ecx = directory buffer sector count
5221                              <1>     ;       (sectors per cluster)
5222                              <1>     ;mul    ecx
5223                              <1>     ;mov    ecx, eax
5224                              <1>             ; (bytes per cluster or directory buffer size)
5225                              <1>     ;shr    ecx, 2 ; dword count
5226                              <1>     ; 19/12/2025
5227 0000BD30 C1E107              <1>     shl     ecx, 7 ; 512/4 (dword count)
5228                              <1>
5229 0000BD33 29C0                <1>     sub     eax, eax ; 0
5230 0000BD35 F3AB                <1>     rep     stosd
5231                              <1>     ;
5232 0000BD37 C605[697E0100]02    <1>     mov     byte [DirBuff_ValidData], 2
5233 0000BD3E E83EF0FFFF          <1>     call    save_directory_buffer
5234 0000BD43 72B8                <1>     jc      short loc_createfile_anc_retn
5235                              <1>
5236                              <1> loc_createfile_save_added_subdir_cluster:
5237 0000BD45 A1[98820100]        <1>     mov     eax, [createfile_LastDirCluster]
5238 0000BD4A 8B0D[94820100]      <1>     mov     ecx, [createfile_FFCluster]
5239 0000BD50 E80B050000          <1>     call    update_cluster
5240 0000BD55 7304                <1>     jnc     short loc_createfile_save_fat_buffer_0
5241 0000BD57 09C0                <1>     or      eax, eax ; EAX = 0 -> cluster value is 0 or eocc
5242 0000BD59 7518                <1>     jnz     short loc_createfile_save_fat_buffer_stc_retn
5243                              <1>
5244                              <1> loc_createfile_save_fat_buffer_0:
5245 0000BD5B A1[94820100]        <1>     mov     eax, [createfile_FFCluster]
5246 0000BD60 A3[98820100]        <1>     mov     [createfile_LastDirCluster], eax
5247                              <1>     ;mov    ecx, 0FFFFFFFh ; 28 bit
5248                              <1>     ; 29/08/2024
5249                              <1>     ; (update_cluster will save 28 bit cluster value)
5250                              <1>     ;mov    ecx, 0FFFFFFFFh
5251 0000BD65 29C9                <1>     sub     ecx, ecx
5252 0000BD67 49                  <1>     dec     ecx ; 0FFFFFFFFh
5253 0000BD68 E8F3040000          <1>     call    update_cluster
5254 0000BD6D 7306                <1>     jnc     short loc_createfile_save_fat_buffer_1
5255 0000BD6F 09C0                <1>     or      eax, eax ; Was it EOF ?
5256 0000BD71 7402                <1>     jz      short loc_createfile_save_fat_buffer_1 ; yes
5257                              <1>
5258                              <1> loc_createfile_save_fat_buffer_stc_retn:
5259 0000BD73 F9                  <1>     stc
5260                              <1> loc_createfile_save_fat_buffer_retn:
5261                              <1> loc_createfile_gffc_2_stc_retn:
5262 0000BD74 C3                  <1>     retn
5263                              <1>
5264                              <1> loc_createfile_save_fat_buffer_1:
5265                              <1>     ; byte [FAT_BuffValidData] = 2
5266 0000BD75 E85A070000          <1>     call    save_fat_buffer
5267 0000BD7A 72F8                <1>     jc      short loc_createfile_save_fat_buffer_retn
5268                              <1>
5269 0000BD7C 803D[5E7E0100]01    <1>     cmp     byte [FAT_ClusterCounter], 1
5270 0000BD83 7222                <1>     jb      short loc_createfile_save_fat_buffer_2
5271                              <1>
5272                              <1>     ; ESI = Logical DOS Drive Description Table address
5273 0000BD85 A1[5E7E0100]        <1>     mov     eax, [FAT_ClusterCounter]
5274                              <1>
5275 0000BD8A C605[5E7E0100]00    <1>     mov     byte [FAT_ClusterCounter], 0 ; 21/03/2016
5276                              <1>
5277 0000BD91 66BB01FF            <1>     mov     bx, 0FF01h ; add free clusters
5278 0000BD95 E8C7070000          <1>     call    calculate_fat_freespace
5279                              <1>
5280                              <1>     ;inc    eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
5281                              <1>     ;jnz    short loc_createfile_save_fat_buffer_2
5282                              <1>
5283                              <1>     ; ecx > 0 -> Recalculation is needed
5284 0000BD9A 09C9                <1>     or      ecx, ecx
5285 0000BD9C 7409                <1>     jz      short loc_createfile_save_fat_buffer_2
5286                              <1>
5287 0000BD9E 66BB00FF            <1>     mov     bx, 0FF00h ; recalculate free space
5288 0000BDA2 E8BA070000          <1>     call    calculate_fat_freespace
5289                              <1>
5290                              <1> loc_createfile_save_fat_buffer_2:
5291                              <1>     ;call   update_parent_dir_lmdt
5292                              <1>
5293                              <1>     ; 27/08/2024
5294                              <1> ;loc_createfile_gffc_2:
5295                              <1>     ;call   get_first_free_cluster
5296                              <1>     ;jc     short loc_createfile_gffc_2_stc_retn
5297                              <1>     ;
5298                              <1>     ;mov    [createfile_FFCluster], eax
5299                              <1>     ;
5300                              <1>
5301 0000BDA7 A1[98820100]        <1>     mov     eax, [createfile_LastDirCluster]
5302                              <1>
5303 0000BDAC E884030000          <1>     call    load_FAT_sub_directory
5304 0000BDB1 72C1                <1>     jc      short loc_createfile_gffc_2_stc_retn
5305                              <1>
5306 0000BDB3 BF00000800          <1>     mov     edi, Directory_Buffer
5307                              <1>
5308                              <1>     ; 27/08/2024
5309                              <1>     ;sub    bx, bx ; directory entry index/number = 0
5310 0000BDB8 29DB                <1>     sub     ebx, ebx
5311                              <1>     ; 29/07/2022
5312                              <1>     ;sub    ecx, ecx
5313                              <1>
```

```
5314 0000BDBA 56                  <1>     push    esi ; * ; 23/03/2016
5315                              <1>
5316                              <1> loc_createfile_set_ff_dir_entry:
5317                              <1>     ;;;
5318                              <1>     ; 29/08/2024
5319 0000BDBB A1[6E7E0100]        <1>     mov     eax, [DirBuff_Cluster]
5320 0000BDC0 A3[98820100]        <1>     mov     [createfile_LastDirCluster], eax
5321                              <1>     ;;;
5322                              <1>
5323                              <1>     ; 27/08/2024
5324 0000BDC5 66891D[A6820100]    <1>     mov     [createfile_DirIndex], bx
5325                              <1>     ; 29/07/2022
5326                              <1>     ;mov    [createfile_DirIndex], cx ; 0
5327                              <1>
5328                              <1>     ;;;
5329 0000BDCC A1[90820100]        <1>     mov     eax, [createfile_size]
5330 0000BDD1 09C0                <1>     or      eax, eax
5331 0000BDD3 740A                <1>     jz      short loc_createfile_sffc
5332                              <1>     ; 31/08/2024
5333 0000BDD5 8B3424              <1>     mov     esi, [esp] ; * ; LDDDT address
5334 0000BDD8 E8AE030000          <1>     call    get_first_free_cluster
5335 0000BDDD 7295                <1>     jc      short loc_createfile_gffc_2_stc_retn
5336                              <1> loc_createfile_sffc:
5337 0000BDDF A3[94820100]        <1>     mov     [createfile_FFCluster], eax
5338 0000BDE4 29C9                <1>     sub     ecx, ecx ; 0
5339                              <1>     ;;;
5340                              <1>
5341                              <1>         ; EDI = Directory entry address
5342 0000BDE6 8B35[88820100]      <1>     mov     esi, [createfile_Name_Offset]
5343                              <1>     ;;;
5344                              <1>     ; 27/08/2024
5345                              <1>     ;mov    eax, [createfile_FFCluster]
5346                              <1>     ;;;
5347 0000BDEC A3[9C820100]        <1>     mov     [createfile_Cluster], eax ; 24/03/2016
5348                              <1>     ;mov    ch, 0FFh
5349                              <1>         ; 29/07/2022
5350                              <1>     ; ecx = 0
5351 0000BDF1 FECD                <1>     dec     ch ; 0 -> 0FFh
5352 0000BDF3 8A0D[A4820100]      <1>     mov     cl, [createfile_attrib] ; file attributes
5353                              <1>     ; CH > 0 -> File size is in [EBX]
5354 0000BDF9 BB[90820100]        <1>     mov     ebx, createfile_size
5355                              <1>
5356 0000BDFE E8ABEEFFFF          <1>     call    make_directory_entry
5357                              <1>
5358 0000BE03 5E                  <1>     pop     esi ; * ; ESI = Logical Dos Drv Desc. Table address
5359                              <1>
5360 0000BE04 C605[697E0100]02    <1>     mov     byte [DirBuff_ValidData], 2
5361 0000BE0B E871EFFFFF          <1>     call    save_directory_buffer
5362 0000BE10 7213                <1>     jc      short loc_createfile_set_ff_dir_entry_retn
5363                              <1>
5364 0000BE12 C605[AD820100]01    <1>     mov     byte [createfile_UpdatePDir], 1 ; 31/03/2016
5365                              <1>
5366                              <1> loc_createfile_get_set_write_file_cluster:
5367 0000BE19 A1[90820100]        <1>     mov     eax, [createfile_size]
5368 0000BE1E 09C0                <1>     or      eax, eax ; 0 ?
5369 0000BE20 755B                <1>     jnz     short loc_createfile_get_set_wfc_cont ; no
5370 0000BE22 40                  <1>     inc     eax ; eax = 1
5371                              <1>
5372                              <1>     ; 27/08/2024 (empty file)
5373                              <1>     ;;;
5374                              <1>     ; 23/03/2016
5375                              <1>     ;movzx  ebx, byte [createfile_SecPerClust]
5376                              <1>     ;;movzx ecx, word [esi+LD_BPB+BytesPerSec] ; 512
5377                              <1>        ;movzx     ecx, word [createfile_BytesPerSec] ; 512
5378                              <1>     ;;;
5379 0000BE23 EB72                <1>     jmp     loc_createfile_set_cluster_count
5380                              <1>
5381                              <1> loc_createfile_set_ff_dir_entry_retn:
5382 0000BE25 C3                  <1>     retn
5383                              <1>
5384                              <1> loc_createfile_write_fcluster_to_disk:
5385 0000BE26 034668              <1>     add     eax, [esi+LD_DATABegin] ; convert to physical address
5386 0000BE29 BB00000700          <1>     mov     ebx, Cluster_Buffer
5387                              <1>     ; ESI = Logical DOS Drv. Desc. Tbl. address
5388                              <1>     ; EAX = Disk address
5389                              <1>     ; EBX = Sector Buffer
5390                              <1>     ; ECX = sectors per cluster
5391 0000BE2E E8805B0000          <1>     call    disk_write
5392 0000BE33 7211                <1>     jc      short loc_createfile_dsk_wr_err
5393                              <1>
5394                              <1> loc_createfile_update_fat_cluster:
5395                              <1>     ; 21/03/2016
5396 0000BE35 803D[AC820100]00    <1>     cmp     byte [createfile_wfc], 0
5397 0000BE3C 7711                <1>     ja      short loc_createfile_update_fat_cluster_n1
5398                              <1>
5399 0000BE3E FE05[AC820100]      <1>     inc     byte [createfile_wfc] ; 1
5400 0000BE44 EB1F                <1>     jmp     short loc_createfile_update_fat_cluster_n2
5401                              <1>
5402                              <1> loc_createfile_dsk_wr_err:
5403                              <1>     ; 16/10/2016 (1Dh -> 18)
5404                              <1>     ; 23/03/2016
5405                              <1>     ;mov    eax, 18 ; Drive not ready or write error !
5406                              <1>     ; 29/07/2022
5407 0000BE46 29C0                <1>     sub     eax, eax
5408 0000BE48 B012                <1>     mov     al, 18
5409                              <1> loc_cf_stc_retn:
5410 0000BE4A E9BA000000          <1>     jmp     loc_createfile_stc_retn
5411                              <1>
5412                              <1> loc_createfile_update_fat_cluster_n1:
5413 0000BE4F A1[A0820100]        <1>     mov     eax, [createfile_PCluster]
5414 0000BE54 8B0D[9C820100]      <1>     mov     ecx, [createfile_Cluster]
5415 0000BE5A E801040000          <1>     call    update_cluster
5416 0000BE5F 7304                <1>     jnc     short loc_createfile_update_fat_cluster_n2
5417 0000BE61 09C0                <1>     or      eax, eax ; EAX = 0 -> cluster value is 0 or eocc
5418                              <1>     ;jnz    loc_createfile_stc_retn
5419                              <1>     ; 29/07/2022
5420 0000BE63 75E5                <1>     jnz     short loc_cf_stc_retn
5421                              <1>
5422                              <1> loc_createfile_update_fat_cluster_n2:
5423 0000BE65 A1[9C820100]        <1>         mov eax, [createfile_Cluster]
5424                              <1>     ;mov    ecx, 0FFFFFFFh
5425                              <1>     ; 29/08/2024
5426                              <1>     ; (update_cluster will save 28 bit cluster value)
5427                              <1>     ;mov    ecx, 0FFFFFFFFh
5428 0000BE6A 29C9                <1>     sub     ecx, ecx
5429 0000BE6C 49                  <1>     dec     ecx
5430 0000BE6D E8EE030000          <1>     call    update_cluster
5431 0000BE72 7353                <1>     jnc     short loc_createfile_save_fat_buffer_3
5432 0000BE74 09C0                <1>     or      eax, eax ; EAX = 0 -> cluster value is 0 or eocc
5433 0000BE76 744F                <1>     jz      short loc_createfile_save_fat_buffer_3
5434                              <1>
5435                              <1> loc_cf_upd_fat_fcluster_stc_retn:
5436 0000BE78 E98C000000          <1>     jmp     loc_createfile_stc_retn
5437                              <1>
```

```
                                <1> loc_createfile_get_set_wfc_cont:
                                <1>     ;;movzx ecx, word [esi+LD_BPB+BytesPerSec] ; 512
                                <1>     ;movzx  ecx, word [createfile_BytesPerSec] ; 512
                                <1>     ; 19/12/2025
0000BE7D B900020000             <1>     mov     ecx, 512
0000BE82 01C8                   <1>     add     eax, ecx
0000BE84 48                     <1>     dec     eax  ; add eax, 511
0000BE85 29D2                   <1>     sub     edx, edx
0000BE87 F7F1                   <1>     div     ecx
0000BE89 0FB61D[A5820100]       <1>     movzx   ebx, byte [createfile_SecPerClust]
0000BE90 01D8                   <1>     add     eax, ebx
0000BE92 48                     <1>     dec     eax  ; add eax, SecPerClust - 1
                                <1>     ;xor   dx, dx
                                <1>     ; 27/08/2024
0000BE93 31D2                   <1>     xor     edx, edx
0000BE95 F7F3                   <1>     div     ebx
                                <1>
                                <1> loc_createfile_set_cluster_count:
0000BE97 A3[A8820100]           <1>     mov     [createfile_CCount], eax
                                <1>
                                <1>     ; 31/08/2024
                                <1>     ;;;
0000BE9C 833D[90820100]00       <1>     cmp     dword [createfile_size], 0 ; empty file ?
0000BEA3 7422                   <1>     jz      short loc_createfile_save_fat_buffer_3
                                <1>     ;;;
                                <1>
0000BEA5 BF00000700             <1>     mov     edi, Cluster_Buffer
0000BEAA 89C8                   <1>     mov     eax, ecx ; Bytes per Sector
0000BEAC F7E3                   <1>     mul     ebx ; Sectors per Cluster
                                <1>     ; EAX = Bytes per Cluster
0000BEAE 89C1                   <1>     mov     ecx, eax
0000BEB0 C1E902                 <1>     shr     ecx, 2 ; dword count
0000BEB3 31C0                   <1>     xor     eax, eax
0000BEB5 F3AB                   <1>     rep     stosd ; clear cluster buffer
                                <1>
0000BEB7 A1[9C820100]           <1>     mov     eax, [createfile_Cluster] ; 24/03/2016
                                <1>
0000BEBC 89D9                   <1>     mov     ecx, ebx
                                <1>
                                <1> loc_createfile_get_set_wf_fclust_cont:
                                <1>     ;sub    eax, 2
                                <1>     ; 30/07/2022
0000BEBE 48                     <1>     dec     eax
0000BEBF 48                     <1>     dec     eax
0000BEC0 F7E1                   <1>     mul     ecx
                                <1>     ; EAX = Logical DOS disk address (offset)
0000BEC2 E95FFFFFFF             <1>         jmp     loc_createfile_write_fcluster_to_disk
                                <1>
                                <1> loc_createfile_save_fat_buffer_3:
                                <1>     ; byte [FAT_BuffValidData] = 2
0000BEC7 E808060000             <1>     call    save_fat_buffer
                                <1>     ;jc     loc_createfile_stc_retn
                                <1>     ; 29/07/2022
0000BECC 72AA                   <1>     jc      short loc_cf_upd_fat_fcluster_stc_retn
                                <1>
                                <1>     ; 21/03/2016
                                <1>     ;cmp    byte [FAT_ClusterCounter], 1
                                <1>     ;jb     short loc_createfile_save_fat_buffer_4
                                <1>     ; 31/08/2024
0000BECE 803D[5E7E0100]00       <1>     cmp     byte [FAT_ClusterCounter], 0
0000BED5 761B                   <1>     jna     short loc_createfile_save_fat_buffer_4 ; cf = 0
                                <1>
                                <1>     ; ESI = Logical DOS Drive Description Table address
0000BED7 A1[5E7E0100]           <1>     mov     eax, [FAT_ClusterCounter]
0000BEDC 66BB01FF               <1>     mov     bx, 0FF01h ; add free clusters
0000BEE0 E87C060000             <1>     call    calculate_fat_freespace
                                <1>
                                <1>     ;inc    eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
                                <1>     ;jnz    short loc_createfile_save_fat_buffer_4
                                <1>
                                <1>     ; ecx > 0 -> Recalculation is needed
0000BEE5 09C9                   <1>     or      ecx, ecx
0000BEE7 7409                   <1>     jz      short loc_createfile_save_fat_buffer_4
                                <1>
0000BEE9 66BB00FF               <1>     mov     bx, 0FF00h ; ; recalculate free space
0000BEED E86F060000             <1>     call    calculate_fat_freespace
                                <1>
                                <1> loc_createfile_save_fat_buffer_4:
0000BEF2 FF0D[A8820100]         <1>     dec     dword [createfile_CCount]
                                <1>     ;jz     short loc_createfile_upd_dir_modif_date_time
0000BEF8 743D                   <1>     jz      short loc_createfile_stc_retn_cc ; 31/03/2016
                                <1>
                                <1> loc_createfile_get_set_write_next_cluster:
0000BEFA E88C020000             <1>     call    get_first_free_cluster
0000BEFF 7208                   <1>     jc      short loc_createfile_stc_retn
                                <1>
                                <1> loc_createfile_get_set_write_next_cluster_1:
0000BF01 83F8FF                 <1>     cmp     eax, 0FFFFFFFFh
0000BF04 7211                   <1>     jb      short loc_createfile_get_set_write_next_cluster_2
                                <1>
                                <1> ;loc_createfile_wnc_insufficient_disk_space:
                                <1>     ;mov    eax, 27h ; Insufficient disk space
                                <1>     ; 29/07/2022
                                <1>     ;xor    eax, eax
                                <1>     ; 29/08/2024
                                <1>     ; eax = FFFFFFFFh
0000BF06 40                     <1>     inc     eax ; eax = 0
0000BF07 B027                   <1>     mov     al, 27h
                                <1>
                                <1> loc_createfile_stc_retn:
0000BF09 803D[AC820100]01       <1>     cmp     byte [createfile_wfc], 1
0000BF10 7324                   <1>     jnb     short loc_createfile_err_retn
0000BF12 C3                     <1>     retn
                                <1>
                                <1> loc_createfile_wnc_inv_format_retn:
                                <1>     ;mov    eax, 28
0000BF13 B01C                   <1>     mov     al, 28 ; Invalid format
0000BF15 EBF2                   <1>     jmp     short loc_createfile_stc_retn
                                <1>
                                <1> loc_createfile_get_set_write_next_cluster_2:
0000BF17 83F802                 <1>     cmp     eax, 2
0000BF1A 72F7                   <1>     jb      short loc_createfile_wnc_inv_format_retn
                                <1>
                                <1> loc_createfile_get_set_write_next_cluster_3:
0000BF1C 8B0D[9C820100]         <1>     mov     ecx, [createfile_Cluster]
0000BF22 A3[9C820100]           <1>     mov     [createfile_Cluster], eax
0000BF27 890D[A0820100]         <1>     mov     [createfile_PCluster], ecx
0000BF2D 0FB60D[A5820100]       <1>     movzx   ecx, byte [createfile_SecPerClust]
0000BF34 EB88                   <1>     jmp     short loc_createfile_get_set_wf_fclust_cont
                                <1>
                                <1> loc_createfile_err_retn:
0000BF36 F9                     <1>     stc
                                <1>
                                <1> ;loc_createfile_upd_dir_modif_date_time:
```

```
5562                                <1> loc_createfile_stc_retn_cc: ; 31/03/2016
5563 0000BF37 9C                    <1>     pushf  ; cpu is here for an error return or completion
5564 0000BF38 50                    <1>     push   eax ; error code if cf = 1
5565                                <1>
5566                                <1>     ;call  update_parent_dir_lmdt
5567                                <1>
5568                                <1> ;loc_createfile_stc_retn_cc:
5569 0000BF39 A1[5E7E0100]          <1>     mov    eax, [FAT_ClusterCounter]
5570 0000BF3E 09C0                  <1>     or     eax, eax
5571 0000BF40 741A                  <1>     jz     short loc_createfile_stc_retn_pop_eax
5572 0000BF42 8A3D[42770100]        <1>     mov    bh, [Current_Drv]
5573 0000BF48 B301                  <1>     mov    bl, 01h ; BL = 1 -> add clusters
5574                                <1>     ; NOTE: EAX value will be added to Free Cluster Count
5575                                <1>     ; (If EAX value is negative, Free Cluster Count will be decreased)
5576 0000BF4A E812060000            <1>     call   calculate_fat_freespace
5577                                <1>         ; ESI = Logical DOS Drive Description Table Address
5578                                <1>         ;jc short loc_createfile_stc_retn_pop_eax_cf
5579 0000BF4F 21C9                  <1>     and    ecx, ecx ; cx = 0 -> valid free sector count
5580 0000BF51 7409                  <1>     jz     short loc_createfile_stc_retn_pop_eax
5581                                <1>
5582                                <1> loc_createfile_stc_retn_recalc_FAT_freespace:
5583 0000BF53 66BB00FF              <1>     mov    bx, 0FF00h ; bh = 0FFh ->
5584                                <1>     ; ESI = Logical DOS Drv DT Addr
5585                                <1>     ; BL = 0 -> Recalculate
5586 0000BF57 E805060000            <1>     call   calculate_fat_freespace
5587                                <1>
5588                                <1> loc_createfile_stc_retn_pop_eax:
5589 0000BF5C 58                    <1>     pop    eax
5590 0000BF5D 9D                    <1>     popf
5591 0000BF5E 7218                  <1>     jc     short loc_createfile_retn
5592                                <1>
5593                                <1> loc_createfile_retn_fcluster:
5594 0000BF60 A1[94820100]          <1>     mov    eax, [createfile_FFCluster]
5595                                <1>     ;mov   ebx, createfile_size
5596                                <1>     ;;movzx ecx, byte [esi+LD_BPB+SecPerClust]
5597                                <1>     ;movzx  ecx, byte [createfile_SecPerClust] ; 23/03/2016
5598                                <1>     ;movzx  edx, word [createfile_DirIndex]
5599                                <1>     ; 29/08/2024 - TRDOS 386 v2.0.9
5600 0000BF65 8B1D[90820100]        <1>     mov    ebx, [createfile_size]
5601 0000BF6B 0FB70D[A6820100]      <1>     movzx  ecx, word [createfile_DirIndex] ; = directory entry index
5602 0000BF72 8B15[98820100]        <1>     mov    edx, [createfile_LastDirCluster] ; = [DirBuff_Cluster]
5603                                <1>     ; esi = Logical DOS Drive Description Table address
5604                                <1>     ; byte [esi] = drive name ; 'A','B','C','D'
5605                                <1>
5606                                <1> loc_createfile_retn:
5607 0000BF78 C3                    <1>     retn
5608                                <1>
5609                                <1> ; 19/12/2025 (TRDOS 386 Kernel v2.0.10)
5610                                <1> ; 28/07/2022 (TRDOS 386 Kernel v2.0.5)
5611                                <1> ;
5612                                <1> ;create_fs_file:
5613                                <1> ;   ; temporary (21/03/2016)
5614                                <1> ;   ;retn
5615                                <1> ;
5616                                <1> ;delete_fs_file:
5617                                <1> ;   ; temporary (28/02/2016)
5618                                <1> ;   ;retn
5619                                <1> ;
5620                                <1> ;rename_fs_file_or_directory:
5621                                <1> ;   ;retn
5622                                <1> ;
5623                                <1> ;make_fs_directory:
5624                                <1> ;   ; temporary (21/02/2016)
5625                                <1> ;   ;retn
5626                                <1> ;
5627                                <1> ;add_new_fs_section:
5628                                <1> ;   ; temporary (11/03/2016)
5629                                <1> ;   ;retn
5630                                <1> ;
5631                                <1> ;delete_fs_directory_entry:
5632                                <1> ;   ; temporary (11/03/2016)
5633                                <1> ;   ;retn
5634                                <1> ;
5635                                <1> ;csftdf2_read_fs_file_sectors:
5636                                <1> ;   ; temporary (19/03/2016)
5637                                <1> ;   ;retn
5638                                <1> ;
5639                                <1> ;csftdf2_write_fs_file_sectors:
5640                                <1> ;   ; temporary (19/03/2016)
5641                                <1> ;   retn
3436                                    %include 'trdosk5.s' ; 24/01/2016
   1                                <1> ; ****************************************************************************
   2                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.10) - File System Procedures : trdosk5s
   3                                <1> ; ----------------------------------------------------------------------------
   4                                <1> ; Last Update: 21/12/2025 (Previous: 31/08/2024, v2.0.9)
   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>     ; 07/08/2022
  17                                <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
  18                                <1>     ; 15/10/2016
  19                                <1>     ; 23/03/2016
  20                                <1>     ; 01/02/2016 (TRDOS 386 = TRDOS v2.0)
  21                                <1>     ; 05/07/2011
  22                                <1>     ; 07/07/2009
  23                                <1>     ; 2005
  24                                <1>     ; INPUT ->
  25                                <1>     ;       EAX = Cluster Number (32 bit)
  26                                <1>     ;       ESI = Logical DOS Drive Parameters Table
  27                                <1>     ; OUTPUT ->
  28                                <1>     ;       cf = 0 -> No Error, EAX valid
  29                                <1>     ;       cf = 1 & EAX = 0 -> End Of Cluster Chain
  30                                <1>     ;       cf = 1 & EAX > 0 -> Error
  31                                <1>     ;       ECX = Current/Previous cluster (if CF = 0)
  32                                <1>     ;       EAX = Next Cluster Number (32 bit)
  33                                <1>     ;
  34                                <1>     ; (Modified registers: EAX, ECX, EBX, EDX)
  35                                <1>
  36 0000BF79 A3[527E0100]          <1>     mov    [FAT_CurrentCluster], eax
  37                                <1> check_next_cluster_fat_type:
  38 0000BF7E 29D2                  <1>     sub    edx, edx ; 0
  39 0000BF80 807E0302              <1>     cmp     byte [esi+LD_FATType], 2
  40 0000BF84 7243                  <1>     jb     short get_FAT12_next_cluster
  41                                <1>     ;ja    get_FAT32_next_cluster
  42                                <1>     ; 25/07/2022
  43 0000BF86 7605                  <1>     jna    short get_FAT16_next_cluster
```

```
0000BF88 E9B2000000          <1>     jmp     get_FAT32_next_cluster
                             <1>
                             <1> get_FAT16_next_cluster:
0000BF8D BB00030000          <1>     mov     ebx, 300h ;768
0000BF92 F7F3                <1>     div     ebx
                             <1>     ; EAX = Count of 3 FAT sectors
                             <1>     ; EDX = Cluster Offset (< 768)
                             <1>     ;shl    dx, 1 ; Multiply by 2
                             <1>     ; 25/07/2022
0000BF94 D1E2                <1>     shl     edx, 1
0000BF96 89D3                <1>     mov     ebx, edx ; Byte Offset
0000BF98 81C3001C0900        <1>     add     ebx, FAT_Buffer
0000BF9E 66BA0300            <1>     mov     dx, 3
0000BFA2 F7E2                <1>     mul     edx
                             <1>     ; EAX = FAT Sector (<= 256)
                             <1>     ; EDX = 0
0000BFA4 8A0E                <1>     mov     cl, [esi+LD_Name]
                             <1>     ;cmp    byte [FAT_BuffValidData], 0
0000BFA6 3815[567E0100]      <1>         cmp [FAT_BuffValidData], dl ; 0
0000BFAC 7674                <1>     jna     short load_FAT_sectors0
0000BFAE 3A0D[577E0100]      <1>     cmp     cl, [FAT_BuffDrvName]
0000BFB4 756C                <1>         jne short load_FAT_sectors0
0000BFB6 3B05[5A7E0100]      <1>     cmp     eax, [FAT_BuffSector]
0000BFBC 756A                <1>         jne short load_FAT_sectors1
                             <1>     ;movzx  eax, word [ebx]
0000BFBE 668B03              <1>     mov     ax, [ebx]
                             <1>     ; 01/02/2016
                             <1>     ; DRV_FAT.ASM (21/08/2011) had a FATal bug here !
                             <1>     ; (cmp ah, 0Fh) ! (ax >= FF7h)
                             <1>     ; (how can i do a such mistake!?)
                             <1>     ;cmp    al, 0F7h
                             <1>     ;jb     short loc_pass_gnc_FAT16_eoc_check
                             <1>     ;cmp    ah, 0FFh
                             <1>     ;jb     short loc_pass_gnc_FAT16_eoc_check
0000BFC1 6683F8F7            <1>     cmp     ax, 0FFF7h
0000BFC5 724E                <1>     jb      short loc_pass_gnc_FAT16_eoc_check
                             <1>     ; ax >= FFF7h (cluster 0002h to FFF6h is valid, in use)
0000BFC7 EB4A                <1>     jmp     short loc_pass_gnc_FAT16_eoc_check_xor_eax
                             <1>
                             <1> get_FAT12_next_cluster:
0000BFC9 BB00040000          <1>     mov     ebx, 400h ;1024
0000BFCE F7F3                <1>     div     ebx
                             <1>     ; EAX = Count of 3 FAT sectors
                             <1>     ; EDX = Cluster Offset (< 1024)
                             <1>     ; 25/07/2022
                             <1>     ;push   ax
0000BFD0 50                  <1>     push    eax
                             <1>     ;mov    ax, 3
0000BFD1 B003                <1>     mov     al, 3
                             <1>     ;mul    dx      ; Multiply by 3
0000BFD3 F7E2                <1>     mul     edx
                             <1>     ;shr    ax, 1   ; Divide by 2
0000BFD5 D1E8                <1>         shr eax, 1
                             <1>     ;mov    bx, ax  ; Byte Offset
0000BFD7 89C3                <1>     mov     ebx, eax
0000BFD9 81C3001C0900        <1>     add     ebx, FAT_Buffer
0000BFDF 58                  <1>     pop     eax
                             <1>     ;pop    ax
                             <1>     ;mov    dx, 3
0000BFE0 B203                <1>     mov     dl, 3
0000BFE2 F7E2                <1>     mul     edx
                             <1>     ; EAX = FAT Sector (<= 12)
                             <1>     ; EDX = 0
0000BFE4 8A0E                <1>     mov     cl, [esi+LD_Name]
                             <1>     ;cmp    byte [FAT_BuffValidData], 0
0000BFE6 3815[567E0100]      <1>     cmp     [FAT_BuffValidData], dl ; 0
0000BFEC 7634                <1>     jna     short load_FAT_sectors0
0000BFEE 3A0D[577E0100]      <1>     cmp     cl, [FAT_BuffDrvName]
0000BFF4 752C                <1>     jne     short load_FAT_sectors0
0000BFF6 3B05[5A7E0100]      <1>     cmp     eax, [FAT_BuffSector]
0000BFFC 752A                <1>     jne     short load_FAT_sectors1
0000BFFE A1[527E0100]        <1>     mov     eax, [FAT_CurrentCluster]
                             <1>     ;shr    ax, 1
                             <1>     ; 25/07/2022
0000C003 D1E8                <1>     shr     eax, 1
                             <1>     ;movzx  eax, word [ebx]
0000C005 668B03              <1>     mov     ax, [ebx]
0000C008 7313                <1>     jnc     short get_FAT12_nc_even
                             <1>     ;shr    ax, 4
0000C00A C1E804              <1>     shr     eax, 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
0000C00D 663DF70F            <1>     cmp     ax, 0FF7h
0000C011 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:
0000C013 31C0                <1>     xor     eax, eax ; 0
                             <1> loc_pass_gnc_FAT16_eoc_check:
                             <1> loc_pass_gnc_FAT32_eoc_check:
0000C015 8B0D[527E0100]      <1>     mov     ecx, [FAT_CurrentCluster]
0000C01B F5                  <1>     cmc
0000C01C C3                  <1>     retn
                             <1>
                             <1> get_FAT12_nc_even:
0000C01D 80E40F              <1>     and     ah, 0Fh
0000C020 EBEB                <1>     jmp     short loc_gnc_fat12_eoc_check
                             <1>
                             <1> load_FAT_sectors0:
0000C022 880D[577E0100]      <1>     mov     [FAT_BuffDrvName], cl
                             <1> load_FAT_sectors1:
                             <1>     ; 25/07/2022
                             <1>     ;sub    edx, edx
                             <1>     ; edx = 0
0000C028 A3[5A7E0100]        <1>     mov     [FAT_BuffSector], eax
0000C02D 89C3                <1>     mov     ebx, eax
0000C02F 034660              <1>         add eax, [esi+LD_FATBegin]
0000C032 B202                <1>     mov     dl, 2
                             <1>     ;cmp    byte [esi+LD_FATType], 2
0000C034 385603              <1>         cmp [esi+LD_FATType], dl ; 2
0000C037 7748                <1>     ja      short load_FAT_sectors3
0000C039 0FB74E1C            <1>     movzx   ecx, word [esi+LD_BPB+BPB_FATSz16]
0000C03D EB45                <1>     jmp     short load_FAT_sectors4
                             <1>
                             <1>     ; 07/08/2022
                             <1> get_FAT32_next_cluster:
0000C03F BB80010000          <1>     mov     ebx, 180h ;384
0000C044 F7F3                <1>     div     ebx
                             <1>     ; EAX = Count of 3 FAT sectors
                             <1>     ; EDX = Cluster Offset (< 384)
                             <1>     ;shl    dx, 2   ; Multiply by 4
```

```
                                <1>     ; 25/07/2022
0000C046 C1E202                 <1>     shl     edx, 2
0000C049 89D3                   <1>     mov     ebx, edx ; Byte Offset
0000C04B 81C3001C0900           <1>     add     ebx, FAT_Buffer
0000C051 66BA0300               <1>     mov     dx, 3
0000C055 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
0000C057 8A0E                   <1>     mov     cl, [esi+LD_Name]
                                <1>     ;cmp    byte [FAT_BuffValidData], 0
0000C059 3815[567E0100]         <1>     cmp     [FAT_BuffValidData], dl ; 0
0000C05F 76C1                   <1>     jna     short load_FAT_sectors0
0000C061 3A0D[577E0100]         <1>     cmp     cl, [FAT_BuffDrvName]
0000C067 75B9                   <1>     jne     short load_FAT_sectors0
0000C069 3B05[5A7E0100]         <1>     cmp     eax, [FAT_BuffSector] ; 0, 3, 6, 9 ...
0000C06F 75B7                   <1>     jne     short load_FAT_sectors1
0000C071 8B03                   <1>     mov     eax, [ebx]
0000C073 25FFFFFF0F             <1>     and     eax, 0FFFFFFFh ; 28 bit Cluster
0000C078 3DF7FFFF0F             <1>     cmp     eax, 0FFFFFF7h
0000C07D 7296                   <1>     jb      short loc_pass_gnc_FAT32_eoc_check
                                <1>     ; eax >= FFFFFF7h (cluster 0002h to FFFFFF6h is valid)
0000C07F EB92                   <1>     jmp     short loc_pass_gnc_FAT16_eoc_check_xor_eax
                                <1>
                                <1> load_FAT_sectors3:
0000C081 8B4E2A                 <1>     mov     ecx, [esi+LD_BPB+BPB_FATSz32]
                                <1> load_FAT_sectors4:
0000C084 29D9                   <1>     sub     ecx, ebx ; [FAT_BuffSector]
                                <1>     ; 25/07/2022
0000C086 FEC2                   <1>     inc     dl
                                <1>     ; edx = 3
                                <1>         ;cmp       ecx, 3
0000C088 39D1                   <1>         cmp ecx, edx ; 3
0000C08A 7602                   <1>     jna     short load_FAT_sectors5
                                <1>     ;mov    ecx, 3
                                <1>     ; 25/07/2022
0000C08C 89D1                   <1>     mov     ecx, edx ; 3
                                <1> load_FAT_sectors5:
0000C08E BB001C0900             <1>     mov     ebx, FAT_Buffer
0000C093 E82A590000             <1>     call    disk_read
0000C098 730C                   <1>     jnc     short load_FAT_sectors_ok
                                <1>     ; 15/10/2016 (15h -> 17)
                                <1>     ; 23/03/2016 (15h)
0000C09A B811000000             <1>     mov     eax, 17 ; Drive not ready or read error
                                <1>     ;mov    byte [FAT_BuffValidData], 0
                                <1>     ; 25/07/2022
0000C09F 8825[567E0100]         <1>     mov     byte [FAT_BuffValidData], ah ; 0
0000C0A5 C3                     <1>     retn
                                <1> load_FAT_sectors_ok:
0000C0A6 C605[567E0100]01       <1>     mov     byte [FAT_BuffValidData], 1
0000C0AD A1[527E0100]           <1>     mov     eax, [FAT_CurrentCluster]
0000C0B2 E9C7FEFFFF             <1>         jmp     check_next_cluster_fat_type
                                <1>
                                <1> load_FAT_root_directory:
                                <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <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>
0000C0B7 8A1E                   <1>     mov     bl, [esi+LD_Name]
0000C0B9 8A7E03                 <1>     mov     bh, [esi+LD_FATType]
                                <1>
                                <1>     ;mov    [DirBuff_DRV], bl
                                <1>     ;mov    [DirBuff_FATType], bh
0000C0BC 66891D[677E0100]       <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
0000C0C3 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>
                                <1>     ; 25/07/2022
0000C0C7 89D0                   <1>     mov     eax, edx
0000C0C9 6681FA0002             <1>     cmp     dx, 512 ; Number of Root Dir Entries
0000C0CE 740B                   <1>     je      short lrd_mov_ecx_32
                                <1>     ;mov    eax, edx ; 25/07/2022
                                <1>     ; 23/10/2016
0000C0D0 89C1                   <1>     mov     ecx, eax
0000C0D2 6683C10F               <1>     add     cx, 15 ; round up
                                <1>     ;shr    cx, 4  ; 16 entries per sector (512/32)
                                <1>     ; 25/07/2022
0000C0D6 C1E904                 <1>     shr     ecx, 4
                                <1>     ; ecx = Root directory size in sectors
                                <1>     ;;shl   ax, 5 ; Root directory size in bytes
                                <1>     ; 25/07/2022
                                <1>     ;shl    eax, 5
                                <1>     ;;dec   dx    ; Last entry number of root dir
                                <1>     ;dec    edx
                                <1>     ; cx = Dir Buffer sector count
0000C0D9 EB04                   <1>     jmp     short lrd_check_dir_buffer
                                <1>
                                <1> lrd_mov_ecx_32:
                                <1>     ;mov    ecx, 32
                                <1>     ; 25/07/2022
0000C0DB 29C9                   <1>     sub     ecx, ecx
0000C0DD B120                   <1>     mov     cl, 32
                                <1>     ;dec    dx ; 511
                                <1>     ;mov    ax, 32*512
                                <1>
                                <1> lrd_check_dir_buffer:
```

```
                                <1>     ; 25/07/2022
0000C0DF 4A                     <1>     dec     edx ; root dir entries - 1
0000C0E0 C1E005                 <1>     shl     eax, 5 ; * 32
                                <1>     ;
0000C0E3 29DB                   <1>     sub     ebx, ebx ; 0
0000C0E5 881D[697E0100]         <1>     mov     [DirBuff_ValidData], bl ; 0
0000C0EB 668915[6C7E0100]       <1>     mov     [DirBuff_LastEntry], dx
0000C0F2 891D[6E7E0100]         <1>     mov     [DirBuff_Cluster], ebx ; 0
0000C0F8 66A3[727E0100]         <1>     mov     [DirBuffer_Size], ax
                                <1>
0000C0FE 8B4664                 <1>     mov     eax, [esi+LD_ROOTBegin]
                                <1> read_directory:
0000C101 BB00000800             <1>     mov     ebx, Directory_Buffer
0000C106 51                     <1>     push    ecx ; Directory buffer sector count
0000C107 53                     <1>     push    ebx
0000C108 E8B5580000             <1>     call    disk_read
0000C10D 5B                     <1>     pop     ebx
0000C10E 720B                   <1>     jc      short load_DirBuff_error
                                <1>
                                <1> validate_DirBuff_and_return:
0000C110 59                     <1>     pop     ecx ; Number of loaded sectors
0000C111 C605[697E0100]01       <1>     mov     byte [DirBuff_ValidData], 1
0000C118 31C0                   <1>     xor     eax, eax ; 0 = no error
0000C11A C3                     <1>     retn
                                <1>
                                <1> load_DirBuff_error:
0000C11B 89C8                   <1>     mov     eax, ecx ; remaining sectors
0000C11D 59                     <1>     pop     ecx ; sector count
0000C11E 29C1                   <1>     sub     ecx, eax ; Number of loaded sectors
                                <1>     ; 15/10/2016 (15h -> 17)
                                <1>     ;mov    eax, 17 ; DRV NOT READY OR READ ERROR !
                                <1>     ; 25/07/2022
                                <1>     ;sub    eax, eax
0000C120 B011                   <1>     mov     al, 17
0000C122 F9                     <1>     stc
0000C123 C3                     <1>         retn
                                <1>
                                <1> load_FAT32_root_directory:
                                <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <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>
0000C124 8A1E                   <1>     mov     bl, [esi+LD_Name]
0000C126 8A7E03                 <1>     mov     bh, [esi+LD_FATType]
                                <1>
                                <1>     ;mov    [DirBuff_DRV], bl
                                <1>     ;mov    [DirBuff_FATType], bh
0000C129 66891D[677E0100]       <1>     mov     [DirBuff_DRV], bx
                                <1>
                                <1> load_FAT32_root_dir0:
0000C130 8B4632                 <1>     mov     eax, [esi+LD_BPB+FAT32_RootFClust]
0000C133 EB0C                   <1>     jmp     short load_FAT_sub_dir0
                                <1>
                                <1> load_FAT_sub_directory:
                                <1>     ; 19/12/2025 (TRDOS 386 Kernel v2.0.10)
                                <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <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>
0000C135 8A1E                   <1>     mov     bl, [esi+LD_Name]
0000C137 8A7E03                 <1>     mov     bh, [esi+LD_FATType]
                                <1>
                                <1>     ;mov    [DirBuff_DRV], bl
                                <1>     ;mov    [DirBuff_FATType], bh
0000C13A 66891D[677E0100]       <1>     mov     [DirBuff_DRV], bx
                                <1>
                                <1> load_FAT_sub_dir0:
0000C141 0FB64E13               <1>     movzx   ecx, byte [esi+LD_BPB+SecPerClust]
                                <1>
0000C145 882D[697E0100]         <1>     mov     [DirBuff_ValidData], ch ; 0
0000C14B A3[6E7E0100]           <1>     mov     [DirBuff_Cluster], eax
                                <1>
                                <1>     ;movzx  eax, word [esi+LD_BPB+BytesPerSec] ; 512
                                <1>     ;mul    ecx
                                <1>     ;shr    eax, 5 ; directory entry count (dir size / 32)
                                <1>     ; 19/12/2025
0000C150 89C8                   <1>     mov     eax, ecx
0000C152 C1E004                 <1>     shl     eax, 4 ; 512/32 (16 entries per sector)
                                <1>
                                <1>     ;dec    ax ; last entry
                                <1>     ; 25/07/2022
0000C155 48                     <1>     dec     eax
0000C156 66A3[6C7E0100]         <1>     mov     [DirBuff_LastEntry], ax
                                <1>
0000C15C A1[6E7E0100]           <1>     mov     eax, [DirBuff_Cluster]
0000C161 83E802                 <1>     sub     eax, 2
0000C164 F7E1                   <1>     mul     ecx
0000C166 034668                 <1>     add     eax, [esi+LD_DATABegin]
                                <1>     ; ecx = sectors per cluster (dir buffer size <= 128 sectors)
0000C169 EB96                   <1>     jmp     short read_directory
                                <1>
                                <1> ; DRV_FS.ASM
                                <1>
                                <1> ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <1>
                                <1> ; 19/12/2025
                                <1> ;load_current_FS_directory:
                                <1> ;   ;retn
```

```
                            <1> ;load_FS_root_directory:
                            <1> ;   ;retn
                            <1> ;load_FS_sub_directory:
                            <1> ;   retn
                            <1>
                            <1> read_cluster:
                            <1>     ; 19/12/2025 (TRDOS 386 v2.0.10)
                            <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
                            <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>
0000C16B 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
                            <1>
                            <1> ; 19/12/2025
                            <1> %if 0
                            <1>     ;cmp    byte [esi+LD_FATType], 0
                            <1>     ; 25/07/2022
                            <1>     cmp     [esi+LD_FATType], ch ; 0
                            <1>     jna     short read_fs_cluster
                            <1> %endif
                            <1>
                            <1> read_fat_file_sectors: ; 18/03/2016
0000C16F 83E802             <1>     sub     eax, 2 ; Beginning cluster number is always 2
0000C172 0FB65613           <1>     movzx   edx, byte [esi+LD_BPB+BPB_SecPerClust] ; 18/03/2016
0000C176 F7E2               <1>     mul     edx
0000C178 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>
0000C17B E842580000         <1>     call    disk_read
0000C180 7306               <1>     jnc     short rclust_retn
                            <1>
                            <1>     ; 15/10/2016 (15h -> 17)
0000C182 B811000000         <1>     mov     eax, 17 ; Drive not ready or read error !
0000C187 C3                 <1>     retn
                            <1>
                            <1> rclust_retn:
0000C188 29C0               <1>     sub     eax, eax ; 0
0000C18A C3                 <1>     retn
                            <1>
                            <1> ; 19/12/2025
                            <1> %if 0
                            <1>
                            <1> read_fs_cluster:
                            <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
                            <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)
                            <1>     ;mov    ecx, 128 ; maximum count of sectors (before eof)
                            <1>     ; 25/07/2022
                            <1>     sub     ecx, ecx
                            <1>     mov     cl, 128
                            <1>     ;call   read_fs_sectors
                            <1>     ;retn
                            <1>     ;jmp    short read_fs_sectors
                            <1>
                            <1> read_fs_sectors:
                            <1>     ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
                            <1>     stc
                            <1>     retn
                            <1>
                            <1> %endif
                            <1>
                            <1> get_first_free_cluster:
                            <1>     ; 21/12/2025 (TRDOS 386 Kernel v2.0.10)
                            <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
                            <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>     ;
                            <1>     ; (Modified registers: eax, ebx, ecx, edx)
                            <1>     ;
                            <1>
0000C18B 8B4678             <1>     mov     eax, [esi+LD_Clusters]
0000C18E 40                 <1>     inc     eax ; add eax, 1
0000C18F A3[F4800100]       <1>     mov     [gffc_last_free_cluster], eax
                            <1>
0000C194 31DB               <1>     xor     ebx, ebx ; 0 ; 02/03/2016
                            <1>
0000C196 807E0302           <1>     cmp     byte [esi+LD_FATType], 2
0000C19A 760D               <1>     jna     short loc_gffc_get_first_fat_free_cluster0
                            <1>
                            <1> ; 21/12/2025 - TRDOS 386 v2.0.10
```

```
                            <1> %if 1
                            <1>     ; FAT32 - FSINFO - first free cluster
0000C19C 31D2               <1>     xor     edx, edx
0000C19E 4A                 <1>     dec     edx ; -1
0000C19F 87563E             <1>     xchg    edx, [esi+LD_BPB+BPB_Reserved+4]
                            <1>
                            <1> %else
                            <1> loc_gffc_get_first_fat32_free_cluster:
                            <1>     ; 02/03/2016
                            <1>     call    get_fat32_fsinfo_sector_parms
                            <1>     jc      short loc_gffc_get_first_fat_free_cluster0
                            <1> %endif
                            <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)
0000C1A2 89D0               <1>     mov     eax, edx ; FSI_Next_Free (First Free Cluster)
0000C1A4 83F8FF             <1>     cmp     eax, 0FFFFFFFFh ; invalid (unknown) !
0000C1A7 7204               <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:
                            <1>     ;mov    eax, 2
                            <1>     ; 25/07/2022
0000C1A9 29C0               <1>     sub     eax, eax
0000C1AB B002               <1>     mov     al, 2
                            <1>     ;xor    edx, edx
                            <1>
                            <1> loc_gffc_get_first_fat_free_cluster1:
0000C1AD 53                 <1>     push    ebx ; 02/03/2016
                            <1>
                            <1> loc_gffc_get_first_fat_free_cluster2:
0000C1AE A3[F0800100]       <1>     mov     [gffc_first_free_cluster], eax
0000C1B3 A3[EC800100]       <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:
0000C1B8 E8BCFDFFFF         <1>     call    get_next_cluster
0000C1BD 7307               <1>     jnc     short loc_gffc_get_first_fat_free_cluster4
0000C1BF 09C0               <1>     or      eax, eax
0000C1C1 740B               <1>     jz      short loc_gffc_first_free_fat_cluster_next
0000C1C3 5B                 <1>     pop     ebx ; 02/03/2016
0000C1C4 F5                 <1>     cmc     ; stc
0000C1C5 C3                 <1>     retn
                            <1>
                            <1> loc_gffc_get_first_fat_free_cluster4:
0000C1C6 21C0               <1>     and     eax, eax ; next cluster value
0000C1C8 7504               <1>     jnz     short loc_gffc_first_free_fat_cluster_next
0000C1CA 89C8               <1>     mov     eax, ecx ; current (previous cluster) value
0000C1CC EB22               <1>     jmp     short loc_gffc_check_for_set
                            <1>
                            <1> loc_gffc_first_free_fat_cluster_next:
0000C1CE A1[EC800100]       <1>     mov     eax, [gffc_next_free_cluster]
0000C1D3 3B05[F4800100]     <1>     cmp     eax, [gffc_last_free_cluster]
0000C1D9 7308               <1>     jnb     short retn_stc_from_get_first_free_cluster
                            <1> pass_gffc_last_cluster_eax_check:
0000C1DB 40                 <1>     inc     eax ; add eax, 1
0000C1DC A3[EC800100]       <1>     mov     [gffc_next_free_cluster], eax
0000C1E1 EBD5               <1>     jmp     short loc_gffc_get_first_fat_free_cluster3
                            <1>
                            <1> retn_stc_from_get_first_free_cluster:
0000C1E3 A1[F0800100]       <1>     mov     eax, [gffc_first_free_cluster]
0000C1E8 83F802             <1>     cmp     eax, 2
0000C1EB 7712               <1>     ja      short loc_gffc_check_previous_clusters
0000C1ED 29C0               <1>     sub     eax, eax
0000C1EF 48                 <1>     dec     eax ; FFFFFFFFh
                            <1>
                            <1> loc_gffc_check_for_set:
                            <1>     ; 02/03/2016
0000C1F0 5B                 <1>     pop     ebx
                            <1>
                            <1>     ; 21/12/2025
                            <1>     ;;;
0000C1F1 807E0302           <1>     cmp     byte [esi+LD_FATType], 2
0000C1F5 7603               <1>     jna     short loc_gffc_check_for_set_@
                            <1>
0000C1F7 89463E             <1>     mov     [esi+LD_BPB+BPB_Reserved+4], eax
                            <1>
                            <1> loc_gffc_check_for_set_@:
                            <1>     ;;;
                            <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>
0000C1FA 09DB               <1>     or      ebx, ebx
0000C1FC 750D               <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:
0000C1FE C3                 <1>     retn
                            <1>
                            <1> loc_gffc_check_previous_clusters:
0000C1FF 48                 <1>     dec     eax ; sub eax, 1
0000C200 A3[F4800100]       <1>     mov     [gffc_last_free_cluster], eax
                            <1>     ;mov    eax, 2
                            <1>     ; 25/07/2022
0000C205 31C0               <1>     xor     eax, eax
0000C207 B002               <1>     mov     al, 2
                            <1>     ; eax = 2
                            <1>     ;xor    edx, edx
0000C209 EBA3               <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>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
                            <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'
0000C20B 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>
0000C20E 813B52526141           <1>     cmp     dword [ebx], 41615252h
0000C214 753C                   <1>     jne     short loc_sffc_read_fsinfo_sector
0000C216 81BBE4010000727241-    <1>     cmp     dword [ebx+484], 61417272h
0000C21F 61                     <1>
0000C220 7530                   <1>     jne     short loc_sffc_read_fsinfo_sector
                                <1>
0000C222 3B83EC010000           <1>     cmp     eax, [ebx+492]  ; FSI_Next_Free
0000C228 741E                   <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'
0000C22A 8983EC010000           <1>     mov     [ebx+492], eax
0000C230 A1[04810100]           <1>     mov     eax, [CFS_FAT32FSINFOSEC]
                                <1>     ;mov    ecx, 1
                                <1>     ; 25/07/2022
0000C235 31C9                   <1>     xor     ecx, ecx
0000C237 FEC1                   <1>     inc     cl
                                <1>     ; ecx = 1
0000C239 53                     <1>     push    ebx
0000C23A E874570000             <1>     call    disk_write
0000C23F 7208                   <1>     jc      short loc_sffc_read_fsinfo_sector_err1
0000C241 5B                     <1>     pop     ebx
                                <1>
0000C242 8B83EC010000           <1>     mov     eax, [ebx+492] ; First (Next) Free Cluster
                                <1>
                                <1> loc_sffc_retn:
0000C248 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:
                                <1>     ; 25/07/2022
                                <1>     ;mov    ebx, 0
                                <1>     ; 15/10/2016 (1Dh -> 18)
                                <1>     ; 23/03/2016 (1Dh)
                                <1>     ;mov    eax, 18 ; Drive not ready or write error
0000C249 31C0                   <1>     xor     eax, eax
0000C24B 89C3                   <1>     mov     ebx, eax ; 0
0000C24D B012                   <1>     mov     al, 18
0000C24F F9                     <1>     stc
                                <1> loc_sffc_read_fsinfo_sector_err2:
0000C250 5A                     <1>     pop     edx
0000C251 C3                     <1>     retn
                                <1>
                                <1> loc_sffc_read_fsinfo_sector:
0000C252 50                     <1>     push    eax
                                <1>
0000C253 E84F050000             <1>     call    get_fat32_fsinfo_sector_parms
0000C258 72F6                   <1>     jc      short loc_sffc_read_fsinfo_sector_err2
                                <1>
0000C25A 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)
0000C25B 39D0                   <1>     cmp     eax, edx ; First free Cluster (eax = new value)
0000C25D 75CB                   <1>     jne     short loc_sffc_write_fsinfo_sector
                                <1>
0000C25F C3                     <1>     retn
                                <1>
                                <1> update_cluster:
                                <1>     ; 31/08/2024
                                <1>     ; 29/08/2024
                                <1>     ; 26/08/2024
                                <1>     ; 24/08/2024 (TRDOS 386 Kernel v2.0.9)
                                <1>     ; 07/08/2022
                                <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <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>
0000C260 A3[527E0100]       <1>     mov     [FAT_CurrentCluster], eax
0000C265 890D[F8800100]     <1>     mov     [ClusterValue], ecx
                            <1>
                            <1> loc_update_cluster_check_fat_buffer:
0000C26B 8A1E               <1>     mov     bl, [esi+LD_Name]
0000C26D 381D[577E0100]     <1>     cmp     [FAT_BuffDrvName], bl
0000C273 7418               <1>     je      short loc_update_cluster_check_fat_type
0000C275 803D[567E0100]02   <1>     cmp     byte [FAT_BuffValidData], 2
                            <1>     ;je     loc_uc_save_fat_buffer
                            <1>     ; 25/07/2022
0000C27C 7502               <1>     jne     short loc_uc_reset_fat_buffer_validation
0000C27E EB66               <1>     jmp     loc_uc_save_fat_buffer
                            <1>
                            <1> loc_uc_reset_fat_buffer_validation:
0000C280 C605[567E0100]00   <1>     mov     byte [FAT_BuffValidData], 0
                            <1>
                            <1> loc_uc_check_fat_type_reset_drvname:
0000C287 881D[577E0100]     <1>     mov     [FAT_BuffDrvName], bl
                            <1>
                            <1> loc_update_cluster_check_fat_type:
0000C28D 29D2               <1>     sub     edx, edx ; 26/02/2016
0000C28F 8A5E03             <1>     mov     bl, [esi+LD_FATType]
0000C292 83F802             <1>     cmp     eax, 2
0000C295 7218               <1>         jb  short update_cluster_inv_data
                            <1>
                            <1>     ;;;
                            <1>     ; 24/08/2024
                            <1>     ; edx = 0 ; 24/08/2024
0000C297 8B4E78             <1>     mov     ecx, [esi+LD_Clusters]
0000C29A 41                 <1>     inc     ecx
0000C29B 890D[627E0100]     <1>     mov     [LastCluster], ecx
                            <1>     ;
0000C2A1 39C8               <1>     cmp     eax, ecx
                            <1>     ;ja     short return_uc_fat32_stc ; 25/07/2022
                            <1>     ; edx = 0 ; (must be -1 or > 0 after here)
0000C2A3 7717               <1>     ja      short return_uc_fat_stc
                            <1>     ;;;
                            <1>
0000C2A5 80FB02             <1>     cmp     bl, 2
                            <1>         ;ja update_fat32_cluster
                            <1>     ; 25/07/2022
0000C2A8 7608               <1>     jna     short loc_uc_check_fat_type_1
0000C2AA E97B010000         <1>     jmp     update_fat32_cluster
                            <1>
                            <1> ; 24/08/2024
                            <1> ;loc_uc_check_fat_type_1:
                            <1> ;   ;cmp    bl, 1
                            <1> ;   ;jb     short update_cluster_inv_data
                            <1> ;   mov     ecx, [esi+LD_Clusters]
                            <1> ;   inc     ecx
                            <1> ;   mov     [LastCluster], ecx
                            <1> ;   cmp     eax, ecx ; dword [LastCluster]
                            <1> ;   ;ja     return_uc_fat_stc
                            <1> ;   ; 25/07/2022
                            <1> ;   jna     short loc_uc_check_fat_type_2
                            <1> ;   ; 24/08/2024
                            <1> ;   ; edx = 0 ; (must be -1 or > 0 after here)
                            <1> ;   jmp     return_uc_fat_stc
                            <1>
                            <1>     ; 25/07/2022
                            <1> update_cluster_inv_data:
                            <1>     ;;mov   eax, 0Dh
                            <1>     ;mov    al, 0Dh  ; Invalid Data
                            <1>     ; 31/08/2024
0000C2AF B01D               <1>     mov     al, ERR_INV_DATA ; 29 ; Invalid Data
0000C2B1 C3                 <1>     retn
                            <1>
                            <1>     ; 24/08/2024
                            <1> loc_uc_check_fat_type_1:
                            <1> loc_uc_check_fat_type_2:
                            <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:')
0000C2B2 80FB01             <1>     cmp     bl, 1 ; correct comparison is this !
                            <1>     ;jna    update_fat12_cluster
                            <1>     ; 25/07/2022
0000C2B5 770B               <1>     ja      short update_fat16_cluster
0000C2B7 E9CF000000         <1>     jmp     update_fat12_cluster
                            <1>
                            <1>     ; 24/08/2024
                            <1> ;return_uc_fat16_stc:
                            <1>     ; 25/07/2022
                            <1> return_uc_fat_stc:
                            <1>     ; 24/08/2024
                            <1>     ; edx = 0
0000C2BC 4A                 <1>     dec     edx
                            <1>     ; edx = -1 ; 0FFFFFFFFh
                            <1>     ; ecx > 0
                            <1> return_uc_fat16_stc:       ; 24/08/2024
                            <1> loc_fat_buffer_stc_0:      ; 24/08/2024
                            <1>     ; 01/03/2016
0000C2BD 31C0               <1>     xor     eax, eax
0000C2BF F9                 <1>     stc
0000C2C0 EB70               <1>     jmp     short loc_fat_buffer_stc_1
                            <1>
                            <1> update_fat16_cluster:
                            <1> pass_uc_fat16_errc:
                            <1>     ;sub    edx, edx
                            <1>     ; edx = 0
0000C2C2 BB00030000         <1>     mov     ebx, 300h ; 768
0000C2C7 F7F3               <1>     div     ebx
                            <1>     ; EAX = Count of 3 FAT sectors
                            <1>     ; DX = Cluster index in FAT buffer
                            <1>     ;mov    bx, dx
                            <1>     ; 25/07/2022
0000C2C9 89D3               <1>     mov     ebx, edx
                            <1>     ;shl    bx, 1 ; Multiply by 2
                            <1>     ; 25/07/2022
0000C2CB D1E3               <1>     shl     ebx, 1
0000C2CD 66BA0300           <1>     mov     dx, 3
0000C2D1 F7E2               <1>     mul     edx
                            <1>     ; EAX = FAT Sector
                            <1>     ; EDX = 0
                            <1>     ; EBX = Byte offset in FAT buffer
```

```
0000C2D3 8A0D[567E0100]      <1>     mov     cl, [FAT_BuffValidData]
0000C2D9 80F902              <1>     cmp     cl, 2
0000C2DC 7518                <1>     jne     short loc_uc_check_fat16_buff_sector_load
                             <1>
                             <1> loc_uc_check_fat16_buff_sector_save:
0000C2DE 3B05[5A7E0100]      <1>     cmp     eax, [FAT_BuffSector]
                             <1>     ;jne    short loc_uc_save_fat_buffer
                             <1>     ;jmp    short loc_update_fat16_cell
                             <1>     ; 07/08/2022
0000C2E4 741D                <1>     je      short loc_update_fat16_cell
                             <1>     ;jmp    loc_uc_save_fat_buffer
                             <1>
                             <1>     ; 07/08/2022
                             <1> loc_uc_save_fat_buffer:
                             <1>     ; byte [FAT_BuffValidData] = 2
0000C2E6 E8E9010000          <1>     call    save_fat_buffer
0000C2EB 7267                <1>         jc  short loc_fat_sectors_rw_error2
                             <1>     ;mov    byte [FAT_BuffValidData], 1
0000C2ED A1[527E0100]        <1>     mov     eax, [FAT_CurrentCluster]
                             <1>     ;mov    ecx, [ClusterValue]
                             <1>     ;jmp    short loc_update_cluster_check_fat_buffer
0000C2F2 8A1E                <1>     mov     bl, [esi+LD_Name] ; 01/03/2016
0000C2F4 EB8A                <1>         jmp loc_uc_reset_fat_buffer_validation
                             <1>
                             <1> loc_uc_check_fat16_buff_sector_load:
0000C2F6 80F901              <1>     cmp     cl, 1 ; byte [FAT_BuffValidData]
0000C2F9 7560                <1>         jne short loc_uc_load_fat_sectors
0000C2FB 3B05[5A7E0100]      <1>     cmp     eax, [FAT_BuffSector]
0000C301 7558                <1>         jne short loc_uc_load_fat_sectors
                             <1>
                             <1> loc_update_fat16_cell:
                             <1> loc_update_fat16_buffer:
0000C303 81C3001C0900        <1>     add     ebx, FAT_Buffer ; 26/02/2016
                             <1>     ;movzx  eax, word [ebx]
0000C309 668B03              <1>     mov     ax, [ebx]
                             <1>     ; 01/03/2016
0000C30C 89C2                <1>     mov     edx, eax ; old value of the cluster
0000C30E A3[527E0100]        <1>     mov     [FAT_CurrentCluster], eax
0000C313 8B0D[F8800100]      <1>     mov     ecx, [ClusterValue] ; 32 bits
0000C319 66890B              <1>     mov     [ebx], cx ; 16 bits !
                             <1>
0000C31C C605[567E0100]02    <1>     mov     byte [FAT_BuffValidData], 2
                             <1>
0000C323 6683F802            <1>     cmp     ax, 2
0000C327 7294                <1>     jb      short return_uc_fat16_stc
0000C329 3B05[627E0100]      <1>     cmp     eax, [LastCluster]
0000C32F 778C                <1>     ja      short return_uc_fat16_stc
                             <1>
                             <1> loc_fat_buffer_updated:
                             <1>     ; 01/03/2016
0000C331 F8                  <1>     clc
                             <1> loc_fat_buffer_stc_1:
0000C332 9C                  <1>     pushf
0000C333 21C9                <1>     and     ecx, ecx
0000C335 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
0000C337 FF05[5E7E0100]      <1>     inc     dword [FAT_ClusterCounter]
                             <1>
                             <1> loc_fat_buffer_updated_1: ; new value of the cluster > 0
0000C33D 09D2                <1>     or      edx, edx ; 02/03/2016
0000C33F 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
0000C341 FF0D[5E7E0100]      <1>     dec     dword [FAT_ClusterCounter] ; it may be negative number
                             <1>
                             <1> loc_fat_buffer_updated_2:
0000C347 9D                  <1>     popf
0000C348 C3                  <1>     retn
                             <1>
                             <1>     ; 25/07/2022
                             <1> loc_fat_sectors_rw_error1:
                             <1>     ;mov    byte [FAT_BuffValidData], 0
                             <1>     ; 23/10/2016 (15h -> 17)
                             <1>     ; 23/03/2016
0000C349 B811000000          <1>     mov     eax, 17 ; Drive not ready or read error
0000C34E 8825[567E0100]      <1>     mov     [FAT_BuffValidData], ah ; 0
                             <1>
                             <1> loc_fat_sectors_rw_error2:
                             <1>     ;mov    eax, error code
                             <1>     ;mov    edx, 0
0000C354 8B0D[F8800100]      <1>     mov     ecx, [ClusterValue]
0000C35A C3                  <1>     retn
                             <1>
                             <1>     ; 25/07/2022
                             <1> loc_uc_load_fat_sectors:
0000C35B A3[5A7E0100]        <1>     mov     [FAT_BuffSector], eax
                             <1>
                             <1> load_uc_fat_sectors_zero:
0000C360 034660              <1>     add     eax, [esi+LD_FATBegin]
0000C363 BB001C0900          <1>     mov     ebx, FAT_Buffer
0000C368 B903000000          <1>     mov     ecx, 3
0000C36D E850560000          <1>     call    disk_read
0000C372 72D5                <1>     jc      short loc_fat_sectors_rw_error1
                             <1>
0000C374 C605[567E0100]01    <1>         mov     byte [FAT_BuffValidData], 1
0000C37B A1[527E0100]        <1>     mov     eax, [FAT_CurrentCluster]
0000C380 8B0D[F8800100]      <1>     mov     ecx, [ClusterValue]
0000C386 E902FFFFFF          <1>         jmp     loc_update_cluster_check_fat_type
                             <1>
                             <1> update_fat12_cluster:
                             <1> pass_uc_fat12_errc:
                             <1>     ;sub    edx, edx
0000C38B BB00040000          <1>     mov     ebx, 400h ; 1024
0000C390 F7F3                <1>     div     ebx
                             <1>     ; EAX = Count of 3 FAT sectors
                             <1>     ; DX = Cluster index in FAT buffer
                             <1>     ;mov    cx, 3
                             <1>     ; 25/07/2022
0000C392 29C9                <1>     sub     ecx, ecx
0000C394 B103                <1>     mov     cl, 3
                             <1>     ; ecx = 3
                             <1>     ;mov    bx, ax
0000C396 89C3                <1>     mov     ebx, eax
                             <1>     ;mov    ax, cx ; 3
0000C398 89C8                <1>     mov     eax, ecx
                             <1>     ;mul    dx      ; Multiply by 3
0000C39A F7E2                <1>     mul     edx
                             <1>     ;shr    ax, 1  ; Divide by 2
0000C39C D1E8                <1>     shr     eax, 1
                             <1>     ;xchg   bx, ax
0000C39E 93                  <1>     xchg    ebx, eax
```

```
1035                                <1>     ; EAX = Count of 3 FAT sectors
1036                                <1>     ; EBX = Byte Offset in FAT buffer
1037                                <1>     ;mul    cx  ; 3 * AX
1038 0000C39F F7E1                 <1>     mul     ecx ; 3 * EAX
1039                                <1>     ; EAX = FAT Beginning Sector
1040                                <1>     ; EDX = 0
1041 0000C3A1 8A0D[567E0100]       <1>     mov     cl, [FAT_BuffValidData]
1042                                <1>     ; TRDOS v1 has a FATal bug here !
1043                                <1>     ; (it does not have 'cmp cl, 2' instruction here !
1044                                <1>     ;  while 'jne' is existing !)
1045 0000C3A7 80F902               <1>     cmp     cl, 2 ; 2 = dirty buffer (must be written to disk)
1046 0000C3AA 750D                 <1>     jne     short loc_uc_check_fat12_buff_sector_load
1047                                <1>
1048                                <1> loc_uc_check_fat12_buff_sector_save:
1049 0000C3AC 3B05[5A7E0100]       <1>     cmp     eax, [FAT_BuffSector]
1050                                <1>         ;jne       short loc_uc_save_fat_buffer
1051                                <1>     ;jmp    short loc_update_fat12_cell
1052                                <1>     ; 07/08/2022
1053 0000C3B2 7412                 <1>     je      short loc_update_fat12_cell
1054 0000C3B4 E92DFFFFFF           <1>     jmp     loc_uc_save_fat_buffer
1055                                <1>
1056                                <1> loc_uc_check_fat12_buff_sector_load:
1057 0000C3B9 80F901               <1>     cmp     cl, 1 ; byte ptr [FAT_BuffValidData]
1058 0000C3BC 759D                 <1>         jne     short loc_uc_load_fat_sectors
1059 0000C3BE 3B05[5A7E0100]       <1>     cmp     eax, [FAT_BuffSector]
1060 0000C3C4 7595                 <1>     jne     short loc_uc_load_fat_sectors
1061                                <1>     ; 07/08/2022
1062                                <1>     ;je     short loc_update_fat12_cell
1063                                <1>     ;jmp    loc_uc_load_fat_sectors
1064                                <1>
1065                                <1> loc_update_fat12_cell:
1066 0000C3C6 81C3001C0900         <1>     add     ebx, FAT_Buffer ; 26/02/2016
1067                                <1>     ;mov    cx, [FAT_CurrentCluster]
1068                                <1>     ; 25/07/2022
1069 0000C3CC 8B0D[527E0100]       <1>     mov     ecx, [FAT_CurrentCluster]
1070                                <1>     ;shr    cx, 1
1071                                <1>     ; 25/07/2022
1072 0000C3D2 D1E9                 <1>     shr     ecx, 1
1073 0000C3D4 668B03               <1>     mov     ax, [ebx]
1074                                <1>     ;mov    dx, ax
1075 0000C3D7 89C2                 <1>     mov     edx, eax ; 25/07/2022
1076                                <1>     ; 24/08/2024 (*)
1077 0000C3D9 8B0D[F8800100]       <1>     mov     ecx, [ClusterValue] ; 32 bits
1078 0000C3DF 7336                 <1>     jnc     short uc_fat12_nc_even
1079                                <1>
1080                                <1>     ; 26/08/2024
1081                                <1>     ; ODD cluster number
1082                                <1>     ; eax = current value (before updated)
1083                                <1>     ; ecx = new value of the cluster
1084                                <1>     ; of the cluster
1085                                <1>     ; low 4 bit of al is high 4 bit
1086                                <1>     ; of the previous cluster
1087                                <1>     ;       (it must not be overwritten)
1088                                <1>
1089                                <1>     ;and    ax, 0Fh
1090                                <1>     ; 25/07/2022
1091 0000C3E1 240F                 <1>     and     al, 0Fh
1092                                <1>     ; 24/08/2024 (*)
1093                                <1>     ;mov    ecx, [ClusterValue] ; 32 bits
1094                                <1>     ;shl    cx, 4
1095 0000C3E3 C1E104               <1>     shl     ecx, 4
1096                                <1>     ;or     cx, ax
1097 0000C3E6 08C1                 <1>     or      cl, al ; 25/07/2022
1098                                <1>     ;mov    ax, dx
1099 0000C3E8 89D0                 <1>     mov     eax, edx
1100 0000C3EA 66890B               <1>     mov     [ebx], cx  ; 16 bits !
1101                                <1>     ;shr    ax, 4 ; al(bit4..7)+ah(bit0..7)
1102                                <1>     ; 25/07/2022
1103 0000C3ED C1E804               <1>     shr     eax, 4
1104                                <1>
1105                                <1> update_fat12_buffer:
1106                                <1>     ;;;
1107                                <1>     ; 24/08/2024
1108 0000C3F0 8B0D[F8800100]       <1>     mov     ecx, [ClusterValue]
1109                                <1>     ;;;
1110 0000C3F6 A3[527E0100]         <1>     mov     [FAT_CurrentCluster], eax
1111 0000C3FB 89C2                 <1>     mov     edx, eax ; 01/03/2016
1112 0000C3FD C605[567E0100]02     <1>     mov     byte [FAT_BuffValidData], 2
1113 0000C404 6683F802             <1>     cmp     ax, 2
1114 0000C408 724A                 <1>         jb      short return_uc_fat12_stc
1115 0000C40A 3B05[627E0100]       <1>     cmp     eax, [LastCluster]
1116 0000C410 7742                 <1>         ja      short return_uc_fat12_stc
1117 0000C412 E91AFFFFFF           <1>         jmp     loc_fat_buffer_updated
1118                                <1>
1119                                <1> uc_fat12_nc_even:
1120                                <1>     ; 26/08/2024
1121                                <1>     ; EVEN cluster number
1122                                <1>     ; eax = current value (before updated)
1123                                <1>     ; ecx = new value of the cluster
1124                                <1>     ; of the cluster
1125                                <1>     ; high 4 bit of ah is low 4 bit
1126                                <1>     ; of the next cluster (it must not be overwritten)
1127                                <1>
1128 0000C417 662500F0             <1>     and     ax, 0F000h
1129                                <1>     ; 24/08/2024 (*)
1130                                <1>     ;mov    ecx, [ClusterValue] ; 32 bits
1131 0000C41B 80E50F               <1>     and     ch, 0Fh
1132                                <1>     ;or     cx, ax
1133                                <1>     ; 25/07/2022
1134 0000C41E 09C1                 <1>     or      ecx, eax
1135                                <1>     ;mov    ax, dx
1136 0000C420 89D0                 <1>     mov     eax, edx
1137 0000C422 66890B               <1>     mov     [ebx], cx ; 16 bits !
1138 0000C425 80E40F               <1>     and     ah, 0Fh ; al(bit0..7)+ah(bit0..3)
1139 0000C428 EBC6                 <1>     jmp     short update_fat12_buffer
1140                                <1>
1141                                <1> update_fat32_cluster:
1142                                <1>     ; edx = 0 ; 24/08/2024
1143                                <1>     ;mov    ecx, [esi+LD_Clusters]
1144                                <1>     ;inc    ecx
1145                                <1>     ;mov    [LastCluster], ecx
1146                                <1>     ; 24/08/2024
1147                                <1>     ; ecx = [LastCluster]
1148                                <1>
1149                                <1>     ; 24/08/2024
1150                                <1>     ;cmp    eax, ecx
1151                                <1>     ;;ja    short return_uc_fat32_stc ; 25/07/2022
1152                                <1>     ;; 24/08/2024
1153                                <1>     ;; edx = 0 ; (must be -1 or > 0 after here)
1154                                <1>     ;ja     short return_uc_fat_stc
1155                                <1>
1156                                <1> pass_uc_fat32_errc:
1157                                <1>     ;sub    edx, edx
1158 0000C42A BB80010000           <1>     mov     ebx, 180h ; 384
```

```
1159 0000C42F F7F3                <1>     div     ebx
1160                              <1>     ; EAX = Count of 3 FAT sectors
1161                              <1>     ; DX = Cluster index in FAT buffer
1162 0000C431 89D3                <1>     mov     ebx, edx
1163 0000C433 C1E302              <1>     shl     ebx, 2 ; Multiply by 4
1164                              <1>     ;mov    edx, 3
1165                              <1>     ; 25/07/2022
1166                              <1>     ;xor    dh, dh
1167                              <1>     ;mov    dl, 3
1168 0000C436 66BA0300            <1>     mov     dx, 3
1169 0000C43A F7E2                <1>     mul     edx
1170                              <1>     ; EBX = Cluster Offset in FAT buffer
1171                              <1>     ; EAX = FAT Sector
1172                              <1>     ; EDX = 0
1173 0000C43C 8A0D[567E0100]      <1>     mov     cl, [FAT_BuffValidData]
1174 0000C442 80F902              <1>     cmp     cl, 2
1175 0000C445 7512                <1>     jne     short loc_uc_check_fat32_buff_sector_load
1176                              <1>
1177                              <1> loc_uc_check_fat32_buff_sector_save:
1178 0000C447 3B05[5A7E0100]      <1>     cmp     eax, [FAT_BuffSector]
1179                              <1>     ;jne    loc_uc_save_fat_buffer
1180                              <1>     ;jmp    short loc_update_fat32_cell
1181                              <1>     ; 25/07/2022
1182 0000C44D 741C                <1>     je      short loc_update_fat32_cell
1183 0000C44F E992FEFFFF          <1>     jmp     loc_uc_save_fat_buffer
1184                              <1>
1185                              <1> return_uc_fat12_stc:
1186                              <1> return_uc_fat32_stc:
1187                              <1>     ; 24/08/2024
1188 0000C454 E964FEFFFF          <1>     jmp     loc_fat_buffer_stc_0 ; (*)
1189                              <1>     ; 25/07/2022
1190                              <1>     ;sub    eax, eax
1191                              <1>     ;stc
1192                              <1>     ;jmp    loc_fat_buffer_stc_1 ; (*)
1193                              <1>
1194                              <1> loc_uc_check_fat32_buff_sector_load:
1195 0000C459 80F901                <1>     cmp     cl, 1 ; byte [FAT_BuffValidData]
1196                              <1>     ;jne    loc_uc_load_fat_sectors
1197                              <1>     ; 25/07/2022
1198 0000C45C 7508                <1>     jne     short loc_uc_load_fat_sects
1199 0000C45E 3B05[5A7E0100]      <1>     cmp     eax, [FAT_BuffSector]
1200                              <1>     ;jne    loc_uc_load_fat_sectors
1201                              <1>     ; 25/07/2022
1202 0000C464 7405                <1>     je      short loc_update_fat32_cell
1203                              <1> loc_uc_load_fat_sects:
1204 0000C466 E9F0FEFFFF          <1>     jmp     loc_uc_load_fat_sectors
1205                              <1>
1206                              <1> loc_update_fat32_cell:
1207                              <1> loc_update_fat32_buffer:
1208 0000C46B 81C3001C0900        <1>     add     ebx, FAT_Buffer ; 26/02/2016
1209 0000C471 8B03                <1>     mov     eax, [ebx]
1210 0000C473 25FFFFFF0F          <1>     and     eax, 0FFFFFFFh ; 28 bit cluster value
1211                              <1>
1212 0000C478 8B15[527E0100]      <1>     mov     edx, [FAT_CurrentCluster] ; 01/03/2016
1213                              <1>
1214 0000C47E A3[527E0100]        <1>     mov     [FAT_CurrentCluster], eax
1215 0000C483 8B0D[F8800100]      <1>     mov     ecx, [ClusterValue]
1216                              <1>     ;;;
1217                              <1>     ; 29/08/2024
1218 0000C489 81E1FFFFFF0F        <1>     and     ecx, 0FFFFFFFh ; 28 bit cluster value
1219                              <1>     ;;;
1220 0000C48F 890B                <1>     mov     [ebx], ecx ; 29/02/2016
1221                              <1>
1222 0000C491 C605[567E0100]02    <1>     mov     byte [FAT_BuffValidData], 2
1223                              <1>
1224                              <1>     ; 01/03/2016
1225 0000C498 21C0                <1>     and     eax, eax ; was it free cluster ?
1226 0000C49A 7513                <1>     jnz     short loc_upd_fat32_c0
1227                              <1>
1228                              <1>     ;or     ecx, ecx ; it will be left free ?!
1229                              <1>     ;jz     short loc_upd_fat32_c3
1230                              <1>
1231 0000C49C 3B563E              <1>     cmp     edx, [esi+LD_BPB+BPB_Reserved+4] ; First free cluster
1232 0000C49F 751F                <1>     jne     short loc_upd_fat32_c3
1233                              <1>
1234 0000C4A1 3B15[627E0100]      <1>     cmp     edx, [LastCluster]
1235 0000C4A7 7206                <1>     jb      short loc_upd_fat32_c0
1236                              <1>
1237                              <1>     ;mov    edx, 2 ; rewind !
1238                              <1>     ; 25/07/2022
1239 0000C4A9 29D2                <1>     sub     edx, edx
1240 0000C4AB B202                <1>     mov     dl, 2
1241 0000C4AD EB0E                <1>     jmp     short loc_upd_fat32_c2
1242                              <1>
1243                              <1> loc_upd_fat32_c0:
1244 0000C4AF FF463E              <1>     inc     dword [esi+LD_BPB+BPB_Reserved+4] ; set it to next cluster
1245 0000C4B2 EB0C                <1>     jmp     short loc_upd_fat32_c3
1246                              <1>
1247                              <1> loc_upd_fat32_c1:
1248 0000C4B4 09C9                <1>     or      ecx, ecx ; will it be free cluster ?
1249 0000C4B6 7508                <1>     jnz     short loc_upd_fat32_c3
1250                              <1>
1251 0000C4B8 3B563E              <1>     cmp     edx, [esi+LD_BPB+BPB_Reserved+4] ; First free cluster
1252 0000C4BB 7303                <1>     jnb     short loc_upd_fat32_c3
1253                              <1>
1254                              <1> loc_upd_fat32_c2:
1255 0000C4BD 89563E              <1>     mov     [esi+LD_BPB+BPB_Reserved+4], edx
1256                              <1>
1257                              <1> loc_upd_fat32_c3:
1258 0000C4C0 89C2                <1>     mov     edx, eax
1259                              <1>
1260                              <1> loc_upd_fat32_c4:
1261 0000C4C2 83F802              <1>     cmp     eax, 2
1262 0000C4C5 728D                <1>     jb      short return_uc_fat32_stc ; 25/07/2022
1263                              <1>
1264                              <1> pass_uc_fat32_c_zero_check_2:
1265 0000C4C7 3B05[627E0100]      <1>     cmp     eax, [LastCluster]
1266 0000C4CD 7785                <1>     ja      short return_uc_fat32_stc ; 25/07/2022
1267                              <1>
1268 0000C4CF E95DFEFFFF          <1>     jmp     loc_fat_buffer_updated
1269                              <1>
1270                              <1> save_fat_buffer:
1271                              <1>     ; 19/12/2025 (TRDOS 386 v2.0.10)
1272                              <1>     ; 31/08/2024 (TRDOS 386 v2.0.9)
1273                              <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
1274                              <1>     ; 15/10/2016
1275                              <1>     ; 01/03/2016
1276                              <1>     ; 22/02/2016 (TRDOS 386 = TRDOS v2.0)
1277                              <1>     ; 11/08/2011
1278                              <1>     ; 09/02/2005
1279                              <1>     ; INPUT ->
1280                              <1>     ;       None
1281                              <1>     ; OUTPUT ->
1282                              <1>     ;       cf = 0 -> OK.
```

```
1283                                <1>     ;       cf = 1 -> error code in AL (EAX)
1284                                <1>     ;
1285                                <1>     ;       EBX = FAT_Buffer address
1286                                <1>     ;
1287                                <1>     ; (EAX, EDX, ECX, EBX will be modified)
1288                                <1>
1289                                <1>     ;cmp    byte [FAT_BuffValidData], 2
1290                                <1>     ;je     short loc_save_fat_buff
1291                                <1>
1292                                <1> ;loc_save_fat_buffer_retn:
1293                                <1> ;   xor     eax, eax
1294                                <1> ;   retn
1295                                <1>
1296                                <1> loc_save_fat_buff:
1297 0000C4D4 31D2                 <1>     xor     edx, edx
1298 0000C4D6 8A35[577E0100]       <1>     mov     dh, [FAT_BuffDrvName]
1299 0000C4DC 80FE41               <1>     cmp     dh, 'A'
1300 0000C4DF 724E                 <1>     jb      short loc_save_fat_buffer_inv_data_retn
1301 0000C4E1 80EE41               <1>     sub     dh, 'A'
1302 0000C4E4 56                   <1>     push    esi ; *
1303 0000C4E5 BE00010900           <1>         mov     esi, Logical_DOSDisks
1304 0000C4EA 01D6                 <1>     add     esi, edx
1305                                <1>
1306 0000C4EC 8A5603               <1>     mov     dl, [esi+LD_FATType]
1307                                <1> ; 19/12/2025
1308                                <1> %if 0
1309                                <1>     and     dl, dl
1310                                <1>     jz      short loc_save_fat_buffer_inv_data_pop_retn
1311                                <1> %endif
1312 0000C4EF A1[5A7E0100]          <1>     mov     eax, [FAT_BuffSector]
1313 0000C4F4 80FA02               <1>     cmp     dl, 2
1314 0000C4F7 772E                 <1>     ja      short loc_save_fat32_buff
1315                                <1>
1316                                <1> loc_save_fat_12_16_buff:
1317                                <1>     ; 01/03/2016
1318                                <1>     ; TRDOS v1 has a FATal bug here!
1319                                <1>     ; Correct code: mov dx, word ptr [FAT_BuffSector]+2
1320                                <1>     ; (DX:AX in TRDOS v1 -> EAX in TRDOS v2)
1321                                <1>     ;
1322 0000C4F9 0FB74E1C               <1>     movzx   ecx, word [esi+LD_BPB+FATSecs]
1323 0000C4FD 29C1                 <1>     sub     ecx, eax
1324                                <1>     ; TRDOS v1 has a bug here... ('pop esi' was forgotten!)
1325                                <1>     ;jna    short loc_save_fat_buffer_inv_data_retn ; wrong addr!
1326 0000C4FF 762D                 <1>     jna     short loc_save_fat_buffer_inv_data_pop_retn ; correct addr.
1327                                <1>     ; 25/07/2022
1328                                <1>     ;jmp    short loc_save_fat_buffer_check_rs3
1329                                <1>
1330                                <1> loc_save_fat_buffer_check_rs3:
1331                                <1>     ; 25/07/2022
1332 0000C501 29DB                 <1>     sub     ebx, ebx
1333 0000C503 B303                 <1>     mov     bl, 3
1334                                <1>     ;cmp    ecx, 3
1335 0000C505 39D9                 <1>     cmp     ecx, ebx ; 3
1336 0000C507 7602                 <1>     jna     short loc_save_fat_buff_continue
1337 0000C509 89D9                 <1>     mov     ecx, ebx ; mov ecx, 3
1338                                <1> loc_save_fat_buff_continue:
1339 0000C50B BB001C0900           <1>     mov     ebx, FAT_Buffer
1340 0000C510 034660               <1>     add     eax, [esi+LD_FATBegin]
1341 0000C513 51                   <1>     push    ecx
1342 0000C514 E89A540000           <1>     call    disk_write
1343 0000C519 59                   <1>     pop     ecx
1344 0000C51A 7239                 <1>     jc      short loc_save_FAT_buff_write_err
1345                                <1>
1346 0000C51C 807E0302             <1>     cmp     byte [esi+LD_FATType], 2
1347 0000C520 7613                 <1>     jna     short loc_calc_2nd_fat12_16_addr
1348                                <1>
1349                                <1> loc_calc_2nd_fat32_addr:
1350 0000C522 8B462A               <1>     mov     eax, [esi+LD_BPB+FAT32_FAT_Size]
1351 0000C525 EB12                 <1>     jmp     short loc_calc_2nd_fat_addr
1352                                <1>
1353                                <1>     ; 25/07/2022
1354                                <1> loc_save_fat32_buff:
1355 0000C527 8B4E2A               <1>     mov     ecx, [esi+LD_BPB+FAT32_FAT_Size]
1356 0000C52A 29C1                 <1>     sub     ecx, eax
1357 0000C52C 77D3                 <1>     ja      short loc_save_fat_buffer_check_rs3
1358                                <1>
1359                                <1> loc_save_fat_buffer_inv_data_pop_retn:
1360 0000C52E 5E                   <1>     pop     esi ; *
1361                                <1> loc_save_fat_buffer_inv_data_retn:
1362                                <1>     ;mov    eax, 0Dh ; Invalid DATA
1363                                <1>     ; 25/07/2022
1364 0000C52F 29C0                 <1>     sub     eax, eax
1365                                <1>     ;mov    al, 0Dh  ; Invalid Data
1366                                <1>     ; 31/08/2024
1367 0000C531 B01D                 <1>     mov     al, ERR_INV_DATA ; 29 ; Invalid Data
1368 0000C533 F9                   <1>     stc     ; cf = 1
1369 0000C534 C3                   <1>     retn
1370                                <1>
1371                                <1> loc_calc_2nd_fat12_16_addr:
1372 0000C535 0FB7461C             <1>     movzx   eax, word [esi+LD_BPB+FATSecs]
1373                                <1>
1374                                <1> loc_calc_2nd_fat_addr:
1375 0000C539 034660               <1>     add     eax, [esi+LD_FATBegin]
1376 0000C53C 0305[5A7E0100]       <1>     add     eax, [FAT_BuffSector]
1377 0000C542 BB001C0900           <1>     mov     ebx, FAT_Buffer
1378                                <1>     ; ecx = 1 to 3
1379 0000C547 E867540000           <1>     call    disk_write
1380 0000C54C 7207                 <1>     jc      short loc_save_FAT_buff_write_err
1381                                <1>     ; Valid buffer (1 = valid but do not save)
1382 0000C54E C605[567E0100]01     <1>     mov     byte [FAT_BuffValidData], 1
1383                                <1>
1384                                <1> loc_save_FAT_buff_write_err:
1385 0000C555 5E                   <1>     pop     esi ; *
1386 0000C556 BB001C0900           <1>     mov     ebx, FAT_Buffer
1387                                <1>     ; 15/10/2016 (1Dh -> 18)
1388                                <1>     ; 23/03/2016 (1Dh)
1389 0000C55B B812000000           <1>     mov     eax, 18 ; Drive not ready or write error
1390 0000C560 C3                   <1>     retn
1391                                <1>
1392                                <1> calculate_fat_freespace:
1393                                <1>     ; 21/12/2025 (TRDOS 386 Kernel v2.0.10)
1394                                <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
1395                                <1>     ; 23/03/2016
1396                                <1>     ; 02/03/2016
1397                                <1>     ; 01/03/2016
1398                                <1>     ; 29/02/2016
1399                                <1>     ; 22/02/2016 (TRDOS 386 = TRDOS v2.0)
1400                                <1>     ; 30/04/2011
1401                                <1>     ; 03/04/2010
1402                                <1>     ; 2005
1403                                <1>     ; INPUT ->
1404                                <1>     ;       EAX = Cluster count to be added or subtracted
1405                                <1>     ;       If BH = FFh, ESI = TR-DOS Logical Drive Description Table
1406                                <1>     ;       If BH < FFh, BH = TR-DOS Logical Drive Number
```

```
1407                                <1>     ;       BL:
1408                                <1>     ;       0 = Calculate, 1 = Add, 2 = Subtract, 3 = Get (Not Set/Calc)
1409                                <1>     ; OUTPUT ->
1410                                <1>     ;       EAX = Free Space in sectors
1411                                <1>     ;       ESI = Logical Dos Drive Description Table address
1412                                <1>     ;       BH = Logical Dos Drive Number (same with input value of BH)
1413                                <1>     ;       BL = Type of operation (same with input value of BL)
1414                                <1>     ;       ECX = 0 -> valid
1415                                <1>     ;       ECX > 0 -> error or invalid
1416                                <1>     ;       If EAX = FFFFFFFFh, it is 're-calculation needed'
1417                                <1>     ;                                sign due to r/w error
1418                                <1>     ;
1419                                <1>     ; (Modifed registers: eax, ebx, ecx, edx, esi)
1420                                <1>     ;
1421                                <1>
1422 0000C561 66891D[FE800100]     <1>     mov     [CFS_OPType], bx
1423 0000C568 A3[00810100]         <1>     mov     [CFS_CC], eax
1424                                <1>
1425 0000C56D 80FFFF               <1>     cmp     bh, 0FFh
1426 0000C570 740B                 <1>     je      short pass_calculate_freespace_get_drive_dt_offset
1427                                <1>
1428                                <1> loc_calculate_freespace_get_drive_dt_offset:
1429 0000C572 31C0                 <1>     xor     eax, eax
1430 0000C574 88FC                 <1>         mov     ah, bh
1431 0000C576 BE00010900           <1>     mov     esi, Logical_DOSDisks
1432 0000C57B 01C6                 <1>         add     esi, eax
1433                                <1>
1434                                <1> pass_calculate_freespace_get_drive_dt_offset:
1435 0000C57D 08DB                 <1>     or      bl, bl
1436 0000C57F 7436                 <1>     jz      short loc_reset_fcc
1437                                <1>
1438                                <1> loc_get_free_sectors:
1439 0000C581 8B4674               <1>     mov     eax, [esi+LD_FreeSectors]
1440                                <1>
1441                                <1>     ;xor    ecx, ecx
1442                                <1>     ;dec    ecx ; 0FFFFFFFFh
1443                                <1>     ;cmp    eax, ecx ; 29/02/2016
1444                                <1>     ;je     short loc_get_free_sectors_retn ; recalculation is needed!
1445                                <1>
1446                                <1>     ; 23/03/2016
1447 0000C584 8B4E70               <1>     mov     ecx, [esi+LD_TotalSectors]
1448 0000C587 39C1                 <1>     cmp     ecx, eax ; Total sectors must be greater than Free sectors !
1449 0000C589 7707                 <1>     ja      short loc_get_free_sectors_check_optype
1450                                <1>
1451 0000C58B 31C0                 <1>     xor     eax, eax
1452 0000C58D 48                   <1>     dec     eax ; 0FFFFFFFFh  ; recalculation is needed!
1453 0000C58E 894674               <1>     mov     [esi+LD_FreeSectors], eax ; reset (for recalculation)
1454                                <1>
1455                                <1> loc_get_free_sectors_retn:
1456 0000C591 C3                   <1>     retn
1457                                <1>
1458                                <1> loc_get_free_sectors_check_optype:
1459 0000C592 80FB03               <1>     cmp     bl, 3
1460 0000C595 7203                 <1>     jb      short loc_set_fcc_1 ; 25/07/2022
1461                                <1>
1462 0000C597 29C9                 <1>     sub     ecx, ecx ; 0
1463                                <1>
1464 0000C599 C3                   <1>     retn
1465                                <1>
1466                                <1> loc_set_fcc_1:
1467 0000C59A 807E0302             <1>     cmp     byte [esi+LD_FATType], 2
1468                                <1>     ;ja     loc_update_FAT32_fs_info_fcc
1469                                <1>     ; 25/07/2022
1470 0000C59E 7605                 <1>     jna     short loc_set_fcc_2
1471 0000C5A0 E9DD000000           <1>     jmp     loc_update_FAT32_fs_info_fcc
1472                                <1>
1473                                <1> loc_set_fcc_2:
1474                                <1>     ;mov    eax, [esi+LD_FreeSectors]
1475 0000C5A5 0FB64E13               <1>     movzx   ecx, byte [esi+LD_BPB+SecPerClust]
1476 0000C5A9 29D2                 <1>     sub     edx, edx
1477 0000C5AB F7F1                 <1>     div     ecx
1478                                <1>     ;or     dx, dx
1479                                <1>     ;       ; DX -> Remain sectors < SecPerClust
1480                                <1>     ;       ; DX > 0 -> invalid free sector count
1481                                <1>     ;jnz    short loc_reset_fcc
1482                                <1>
1483                                <1> ;pass_set_fcc_div32:
1484 0000C5AD A3[747E0100]         <1>     mov     [FreeClusterCount], eax
1485 0000C5B2 E986000000           <1>         jmp     loc_set_free_sectors_FAT12_FAT16
1486                                <1>
1487                                <1> loc_reset_fcc:
1488 0000C5B7 31C0                 <1>     xor     eax, eax
1489 0000C5B9 A3[747E0100]         <1>     mov     [FreeClusterCount], eax ; 0
1490 0000C5BE 8B5678               <1>     mov     edx, [esi+LD_Clusters]
1491 0000C5C1 42                   <1>     inc     edx
1492 0000C5C2 8915[627E0100]       <1>     mov     [LastCluster], edx
1493                                <1>
1494 0000C5C8 807E0302             <1>     cmp     byte [esi+LD_FATType], 2
1495 0000C5CC 7645                 <1>     jna     short loc_count_free_fat_clusters_0
1496                                <1>
1497 0000C5CE 48                   <1>     dec     eax ; FFFFFFFFh
1498 0000C5CF A3[08810100]         <1>     mov     [CFS_FAT32FC], eax
1499                                <1>
1500                                <1>     ; 29/02/2016
1501 0000C5D4 89463A               <1>     mov     [esi+LD_BPB+BPB_Reserved], eax ; reset
1502 0000C5D7 89463E               <1>     mov     [esi+LD_BPB+BPB_Reserved+4], eax ; reset
1503                                <1>
1504                                <1>     ;mov    eax, 2
1505                                <1>     ; 25/07/2022
1506 0000C5DA 40                   <1>     inc     eax ; eax = 0
1507 0000C5DB B002                 <1>     mov     al, 2
1508                                <1>
1509                                <1> loc_count_fc_next_cluster_0:
1510 0000C5DD 50                   <1>     push    eax
1511 0000C5DE E896F9FFFF           <1>     call    get_next_cluster
1512 0000C5E3 7310                 <1>     jnc     short loc_check_fat32_ff_cluster
1513 0000C5E5 09C0                 <1>     or      eax, eax
1514 0000C5E7 741E                 <1>     jz      short pass_inc_cfs_fcc_0
1515                                <1>
1516                                <1> loc_put_fcc_unknown_sign:
1517 0000C5E9 58                   <1>     pop     eax
1518                                <1>     ; "Free count is Unknown" sign
1519                                <1>     ;mov    dword [FreeClusterCount], 0FFFFFFFFh
1520                                <1>
1521                                <1>     ; 29/02/2016
1522                                <1>     ; Save Free Cluster Count value in FAT32 'BPB_Reserved' area
1523                                <1>     ;mov    [esi+LD_BPB+BPB_Reserved], 0FFFFFFFFh ; unknown!
1524 0000C5EA 8B15[08810100]       <1>     mov     edx, [CFS_FAT32FC] ; First Free Cluster
1525                                <1>     ; Save First Free Cluster value in FAT32 'BPB_Reserved+4' area
1526 0000C5F0 89563E               <1>     mov     [esi+LD_BPB+BPB_Reserved+4], edx
1527                                <1>
1528 0000C5F3 EB7D                 <1>         jmp     loc_put_fcc_invalid_sign
1529                                <1>
1530                                <1> loc_check_fat32_ff_cluster:
```

```
1531 0000C5F5 09C0              <1>     or      eax, eax
1532 0000C5F7 750E              <1>     jnz     short pass_inc_cfs_fcc_0
1533 0000C5F9 58                <1>     pop     eax
1534 0000C5FA A3[08810100]      <1>     mov     [CFS_FAT32FC], eax
1535                            <1>     ;mov    dword [FreeClusterCount], 1
1536 0000C5FF FF05[747E0100]    <1>     inc     dword [FreeClusterCount]
1537 0000C605 EB27              <1>     jmp     short pass_inc_cfs_fcc_1
1538                            <1>
1539                            <1> pass_inc_cfs_fcc_0:
1540 0000C607 58                <1>     pop     eax
1541                            <1>
1542                            <1> pass_inc_cfs_fcc_0c:
1543 0000C608 40                <1>     inc     eax ; add eax, 1
1544 0000C609 3B05[627E0100]    <1>     cmp     eax, [LastCluster]
1545 0000C60F 76CC              <1>     jna     short loc_count_fc_next_cluster_0
1546 0000C611 EB6F              <1>     jmp     short loc_update_FAT32_fs_info_fcc
1547                            <1>
1548                            <1> loc_count_free_fat_clusters_0:
1549                            <1>     ;mov    eax, 2
1550 0000C613 B002              <1>     mov     al, 2
1551                            <1>
1552                            <1> loc_count_fc_next_cluster:
1553 0000C615 50                <1>     push    eax
1554 0000C616 E85EF9FFFF        <1>     call    get_next_cluster
1555 0000C61B 720C              <1>     jc      short loc_count_fcc_stc
1556                            <1>
1557                            <1> loc_count_free_clusters_1:
1558 0000C61D 21C0              <1>     and     eax, eax
1559 0000C61F 750C              <1>     jnz     short pass_inc_cfs_fcc
1560                            <1>
1561 0000C621 FF05[747E0100]    <1>     inc     dword [FreeClusterCount]
1562 0000C627 EB04              <1>     jmp     short pass_inc_cfs_fcc
1563                            <1>
1564                            <1> loc_count_fcc_stc:
1565 0000C629 09C0              <1>     or      eax, eax
1566 0000C62B 75BC              <1>     jnz     short loc_put_fcc_unknown_sign ; 29/02/2016
1567                            <1>
1568                            <1> pass_inc_cfs_fcc:
1569 0000C62D 58                <1>     pop     eax
1570                            <1>
1571                            <1> pass_inc_cfs_fcc_1:
1572 0000C62E 40                <1>     inc     eax ; add eax, 1
1573 0000C62F 3B05[627E0100]    <1>     cmp     eax, [LastCluster]
1574 0000C635 76DE              <1>     jna     short loc_count_fc_next_cluster
1575                            <1>
1576                            <1> loc_set_free_sectors:
1577 0000C637 807E0302          <1>     cmp     byte [esi+LD_FATType], 2
1578 0000C63B 7745              <1>     ja      short loc_update_FAT32_fs_info_fcc
1579                            <1>
1580                            <1> loc_set_free_sectors_FAT12_FAT16:
1581 0000C63D 803D[FE800100]00  <1>     cmp     byte [CFS_OPType], 0
1582 0000C644 761C              <1>     jna     short pass_FAT_add_sub_fcc
1583 0000C646 A1[00810100]      <1>     mov     eax, [CFS_CC]
1584 0000C64B 803D[FE800100]01  <1>     cmp     byte [CFS_OPType], 1
1585 0000C652 7708              <1>     ja      short pass_FAT_add_fcc
1586 0000C654 0105[747E0100]    <1>     add     [FreeClusterCount], eax
1587 0000C65A EB06              <1>     jmp     short pass_FAT_add_sub_fcc
1588                            <1>
1589                            <1> pass_FAT_add_fcc:
1590 0000C65C 2905[747E0100]    <1>     sub     [FreeClusterCount], eax
1591                            <1>
1592                            <1> pass_FAT_add_sub_fcc:
1593 0000C662 0FB64613          <1>     movzx   eax, byte [esi+LD_BPB+SecPerClust]
1594 0000C666 8B15[747E0100]    <1>     mov     edx, [FreeClusterCount]
1595 0000C66C F7E2              <1>     mul     edx
1596                            <1>
1597 0000C66E 31C9              <1>     xor     ecx, ecx
1598 0000C670 EB05              <1>     jmp     short loc_cfs_retn_params
1599                            <1>
1600                            <1> loc_put_fcc_invalid_sign:
1601 0000C672 29C0              <1>             sub     eax, eax ; 0
1602 0000C674 48                <1>     dec     eax ; FFFFFFFFh
1603                            <1> loc_fat32_ffc_recalc_needed:
1604 0000C675 89C1              <1>     mov     ecx, eax
1605                            <1>
1606                            <1> loc_cfs_retn_params:
1607 0000C677 894674            <1>     mov     [esi+LD_FreeSectors], eax
1608 0000C67A 0FB71D[FE800100]  <1>     movzx   ebx, word [CFS_OPType]
1609 0000C681 C3                <1>     retn
1610                            <1>
1611                            <1> loc_update_FAT32_fs_info_fcc:
1612                            <1> loc_check_fcc_FSINFO_op:
1613                            <1>     ; 29/02/2016
1614                            <1>     ; EAX = Free cluster count (before this update) ; value from disk
1615                            <1>     ; EDX = First Free Cluster (before this update) ; value from disk
1616 0000C682 803D[FE800100]01  <1>     cmp     byte [CFS_OPType], 1
1617 0000C689 7221              <1>     jb      short loc_cfs_FAT32_get_rcalc_parms ; 0 = recalculated
1618 0000C68B 7406              <1>     je      short loc_check_fcc_FSINFO_op1 ; 1 = add
1619                            <1> loc_check_fcc_FSINFO_op2: ; subtract
1620 0000C68D F71D[00810100]    <1>     neg     dword [CFS_CC] ; prepare to subtract ; 2 = sub (add negative)
1621                            <1> loc_check_fcc_FSINFO_op1:
1622                            <1>     ; 01/03/2016
1623 0000C693 31D2              <1>     xor     edx, edx ; 0
1624 0000C695 4A                <1>     dec     edx ; 0FFFFFFFFh
1625 0000C696 8B463A            <1>     mov     eax, [esi+LD_BPB+BPB_Reserved]
1626 0000C699 39D0              <1>     cmp     eax, edx
1627                            <1>     ;jnb    short loc_put_fcc_invalid_sign
1628                            <1>     ; 21/12/2025
1629 0000C69B 73D8              <1>     jnb     short loc_fat32_ffc_recalc_needed
1630                            <1>
1631 0000C69D 0305[00810100]    <1>         add     eax, [CFS_CC] ; free cluster count on disk + current count
1632 0000C6A3 72CD              <1>     jc      short loc_put_fcc_invalid_sign
1633                            <1>
1634 0000C6A5 A3[747E0100]      <1>     mov     [FreeClusterCount], eax
1635 0000C6AA EB0E              <1>     jmp     short loc_cfs_write_FSINFO_sector
1636                            <1>
1637                            <1> loc_cfs_FAT32_get_rcalc_parms:
1638 0000C6AC 8B15[08810100]    <1>     mov     edx, [CFS_FAT32FC]
1639 0000C6B2 A1[747E0100]      <1>     mov     eax, [FreeClusterCount]
1640 0000C6B7 89563E            <1>     mov     [esi+LD_BPB+BPB_Reserved+4], edx ; First Free Cluster
1641                            <1> loc_cfs_write_FSINFO_sector:
1642 0000C6BA 89463A            <1>     mov     [esi+LD_BPB+BPB_Reserved], eax ; Free cluster count
1643                            <1>
1644 0000C6BD E8BC000000        <1>     call    set_fat32_fsinfo_sector_parms
1645 0000C6C2 72AE              <1>     jc      short loc_put_fcc_invalid_sign
1646                            <1>
1647                            <1> loc_set_FAT32_free_sectors:
1648                            <1>     ; 29/02/2016
1649                            <1>     ;mov    eax, [FreeClusterCount]
1650                            <1>     ;mov    ecx, eax
1651                            <1>     ;cmp    eax, 0FFFFFFFFh ; Invalid !
1652                            <1>     ;je     short loc_cfs_retn_params
1653                            <1>     ;
1654 0000C6C4 8B0D[747E0100]    <1>     mov     ecx, [FreeClusterCount]
```

```
1655 0000C6CA 0FB64613            <1>     movzx  eax, byte [esi+LD_BPB+SecPerClust]
1656 0000C6CE F7E1                <1>     mul    ecx
1657                              <1>     ; 29/02/2016
1658 0000C6D0 31C9                <1>     xor    ecx, ecx ; 0
1659 0000C6D2 09D2                <1>     or     edx, edx ; 0 ?
1660 0000C6D4 759C                <1>         jnz short loc_put_fcc_invalid_sign ; 25/07/2022
1661 0000C6D6 394670              <1>     cmp    [esi+LD_TotalSectors], eax ; Volume size in sectors
1662 0000C6D9 7697                <1>         jna     short loc_put_fcc_invalid_sign
1663                              <1>     ;
1664                              <1> loc_set_FAT32_free_sectors_ok:
1665 0000C6DB 31D2                <1>     xor    edx, edx ; 0
1666 0000C6DD EB98                <1>         jmp     short loc_cfs_retn_params
1667                              <1>     ;
1668                              <1>
1669                              <1> get_last_cluster:
1670                              <1>     ; 22/10/2016
1671                              <1>     ; 27/02/2016 (TRDOS 386 = TRDOS v2.0)
1672                              <1>     ; 12/06/2010 (DRV_FAT.ASM, 'proc_get_last_custer')
1673                              <1>     ; 06/06/2010
1674                              <1>     ; INPUT ->
1675                              <1>     ;       EAX = First Cluster Number
1676                              <1>     ;       ESI = Logical Dos Drive Parameters Table
1677                              <1>     ; OUTPUT ->
1678                              <1>     ;       cf = 0 -> No Error, EAX is valid
1679                              <1>     ;       cf = 1 -> EAX > 0 -> Error
1680                              <1>     ;       EAX = Last Cluster Number
1681                              <1>     ;        ECX = Previous Cluster -just before the last cluster-
1682                              <1>     ;        ; 22/10/2016
1683                              <1>     ;       [glc_index] = cluster index number of the last cluster
1684                              <1>     ;
1685                              <1>     ; (Modified registers: EAX, ECX, EBX, EDX)
1686                              <1>
1687 0000C6DF 89C1                <1>     mov    ecx, eax
1688                              <1>
1689 0000C6E1 C705[10810100]FFFF- <1>     mov    dword [glc_index], 0FFFFFFFFh ; 22/10/2016
1689 0000C6E9 FFFF                <1>
1690                              <1>
1691                              <1> loc_glc_get_next_cluster_1:
1692 0000C6EB 890D[0C810100]      <1>     mov    [glc_prevcluster], ecx
1693                              <1>     ; 22/10/2016
1694 0000C6F1 FF05[10810100]      <1>     inc    dword [glc_index]
1695                              <1>
1696                              <1> loc_glc_get_next_cluster_2:
1697 0000C6F7 E87DF8FFFF          <1>     call   get_next_cluster
1698                              <1>     ; ecx = current/previous cluster
1699                              <1>     ; eax = next/last cluster
1700 0000C6FC 73ED                <1>     jnc    short loc_glc_get_next_cluster_1
1701                              <1>
1702 0000C6FE 09C0                <1>     or     eax, eax
1703 0000C700 7509                <1>     jnz    short loc_glc_stc_retn
1704                              <1>
1705                              <1>     ; ecx = previous cluster
1706 0000C702 89C8                <1>         mov eax, ecx
1707                              <1>
1708                              <1>     ; previous cluster becomes last cluster (ecx -> eax)
1709                              <1>     ; previous of previous cluster becomes previous cluster (ecx)
1710                              <1>
1711                              <1> loc_glc_prev_cluster_retn:
1712 0000C704 8B0D[0C810100]      <1>     mov    ecx, [glc_prevcluster]
1713 0000C70A C3                  <1>     retn
1714                              <1>
1715                              <1> loc_glc_stc_retn:
1716 0000C70B F5                  <1>     cmc    ;stc
1717 0000C70C EBF6                <1>         jmp short loc_glc_prev_cluster_retn
1718                              <1>
1719                              <1> truncate_cluster_chain:
1720                              <1>     ; 31/08/2024 - TRDOS 386 v2.0.9
1721                              <1>     ; 01/03/2016
1722                              <1>     ; 28/02/2016 (TRDOS 386 = TRDOS v2.0)
1723                              <1>     ; 22/01/2011 (DRV_FAT.ASM, 'proc_truncate_cluster_chain')
1724                              <1>     ; 11/09/2010
1725                              <1>     ; INPUT ->
1726                              <1>     ;       ESI = Logical dos drive description table address
1727                              <1>     ;       EAX = First cluster to be truncated/unlinked
1728                              <1>     ; OUTPUT ->
1729                              <1>     ;       ESI = Logical dos drive description table address
1730                              <1>     ;       ECX = Count of truncated/removed clusters
1731                              <1>     ;       CF = 0 -> EAX = Free sectors
1732                              <1>     ;       CF = 1 -> Error code in EAX (AL)
1733                              <1>
1734                              <1>     ; NOTE: This procedure does not update lm date&time !
1735                              <1>
1736                              <1> loc_truncate_cc:
1737 0000C70E 31C9                <1>     xor    ecx, ecx ; mov ecx, 0
1738                              <1>     ;mov   byte [FAT_BuffValidData], 0
1739 0000C710 890D[5E7E0100]      <1>     mov    [FAT_ClusterCounter], ecx ; 0 ; reset
1740                              <1>
1741                              <1>     ;;;
1742                              <1>     ; 31/08/2024
1743 0000C716 21C0                <1>     and    eax, eax ; 0
1744 0000C718 740D                <1>     jz     short loc_tcc_unlink_zero_cluster  ; zero
1745                              <1>     ;cmp   eax, 0FFFFFF7h
1746 0000C71A 3DFFFFFF0F          <1>     cmp    eax, 0FFFFFFFh ; 28 bit cluster number limit (EOF)
1747 0000C71F 7207                <1>     jb     short loc_tcc_unlink_clusters
1748                              <1>
1749                              <1>     ; (possible FAT32) EOF signature...
1750                              <1>     ;       not a valid cluster number
1751                              <1>     ;
1752                              <1>     ; NOTE: update_cluster returns EOF (if eax > [LastCluster])
1753                              <1>     ;       instead of invalid data error
1754                              <1>
1755 0000C721 B81D000000          <1>     mov    eax, ERR_INV_DATA  ; invalid cluster number
1756 0000C726 F9                  <1>     stc
1757                              <1> loc_tcc_unlink_zero_cluster:      ; nothing to do
1758 0000C727 C3                  <1>     retn
1759                              <1>     ;;;
1760                              <1>
1761                              <1> loc_tcc_unlink_clusters:
1762 0000C728 E833FBFFFF          <1>     call   update_cluster
1763                              <1>     ; EAX = Next Cluster
1764                              <1>     ; ECX = Cluster Value
1765                              <1>     ; Note:
1766                              <1>     ; Returns count of unlinked clusters in
1767                              <1>     ; dword ptr FAT_ClusterCounter
1768 0000C72D 73F9                <1>     jnc    short loc_tcc_unlink_clusters
1769                              <1>
1770                              <1>     ; error or EOF (end of cluster chain) ; 31/08/2024
1771                              <1>
1772                              <1> pass_tcc_unlink_clusters:
1773 0000C72F A2[17810100]        <1>     mov    byte [TCC_FATErr], al
1774 0000C734 803D[567E0100]02    <1>     cmp    byte [FAT_BuffValidData], 2
1775 0000C73B 750E                <1>     jne    short loc_tcc_calculate_FAT_freespace
1776 0000C73D E892FDFFFF          <1>     call   save_fat_buffer
1777 0000C742 7307                <1>     jnc    short loc_tcc_calculate_FAT_freespace
```

```
1778 0000C744 A2[17810100]        <1>     mov     byte [TCC_FATErr], al ; Error
1779                              <1>     ;mov    byte [FAT_BuffValidData], 0
1780                              <1>
1781                              <1>     ; 01/03/2016
1782 0000C749 EB12                <1>     jmp     short loc_tcc_recalculate_FAT_freespace
1783                              <1>
1784                              <1> loc_tcc_calculate_FAT_freespace:
1785 0000C74B A1[5E7E0100]        <1>     mov     eax, [FAT_ClusterCounter] ; signed (+-) number
1786 0000C750 66BB01FF            <1>     mov     bx, 0FF01h ; BH = FFh -> ESI = Dos drv desc. table
1787                              <1>                        ; BL = 1 -> add cluster(s)
1788 0000C754 E808FEFFFF          <1>     call    calculate_fat_freespace
1789 0000C759 21C9                <1>     and     ecx, ecx ; cx = 0 -> valid free sector count
1790 0000C75B 7409                <1>     jz      short pass_truncate_cc_recalc_FAT_freespace
1791                              <1>
1792                              <1> loc_tcc_recalculate_FAT_freespace:
1793 0000C75D 66BB00FF            <1>     mov     bx, 0FF00h ; recalculate !
1794 0000C761 E8FBFDFFFF          <1>     call    calculate_fat_freespace
1795                              <1>
1796                              <1> loc_tcc_calculate_FAT_freespace_err:
1797                              <1> pass_truncate_cc_recalc_FAT_freespace:
1798 0000C766 8B0D[5E7E0100]      <1>     mov     ecx, [FAT_ClusterCounter]
1799                              <1>
1800 0000C76C 803D[17810100]00    <1>     cmp     byte [TCC_FATErr], 0
1801 0000C773 7608                <1>     jna     short loc_tcc_unlink_clusters_retn
1802                              <1>
1803                              <1> loc_tcc_unlink_clusters_error:
1804 0000C775 0FB605[17810100]    <1>     movzx   eax, byte [TCC_FATErr]
1805 0000C77C F9                  <1>     stc
1806                              <1> loc_tcc_unlink_clusters_retn:
1807 0000C77D C3                  <1>     retn
1808                              <1>
1809                              <1> set_fat32_fsinfo_sector_parms:
1810                              <1>     ; 15/10/2016
1811                              <1>     ; 23/03/2016
1812                              <1>     ; 29/02/2016 (TRDOS 386 = TRDOS v2.0)
1813                              <1>     ; INPUT ->
1814                              <1>     ;       ESI = Logical dos drive description table address
1815                              <1>     ;       [esi+LD_BPB+BPB_Reserved] = Free Cluster Count
1816                              <1>     ;       [esi+LD_BPB+BPB_Reserved+4] = First Free Cluster
1817                              <1>     ; OUTPUT ->
1818                              <1>     ;       ESI = Logical dos drive description table address
1819                              <1>     ;       CF = 0 -> OK..
1820                              <1>     ;       CF = 1 -> Error code in EAX (AL)
1821                              <1>     ;
1822                              <1>     ; (Modified registers: EAX, EBX, ECX, EDX)
1823                              <1>
1824 0000C77E E824000000          <1>     call    get_fat32_fsinfo_sector_parms
1825 0000C783 7221                <1>     jc      short update_fat32_fsinfo_sector_retn
1826                              <1>
1827 0000C785 8B463A              <1>     mov     eax, [esi+LD_BPB+BPB_Reserved] ; Free Cluster Count
1828 0000C788 8B563E              <1>     mov     edx, [esi+LD_BPB+BPB_Reserved+4] ; First free Cluster
1829                              <1>
1830                              <1>         ;mov       ebx, DOSBootSectorBuff
1831 0000C78B 8983E8010000        <1>     mov     [ebx+488], eax
1832 0000C791 8993EC010000        <1>     mov     [ebx+492], edx
1833                              <1>
1834 0000C797 A1[04810100]        <1>     mov     eax, [CFS_FAT32FSINFOSEC]
1835 0000C79C B901000000          <1>     mov     ecx, 1
1836 0000C7A1 E80D520000          <1>     call    disk_write
1837                              <1>     ;jnc    short update_fat32_fsinfo_sector_retn
1838                              <1>
1839                              <1>     ; 15/10/2016 (1Dh -> 18)
1840                              <1>     ; 23/03/2016 (1Dh)
1841                              <1>     ;mov    eax, 18 ; Drive not ready or write error
1842                              <1>
1843                              <1> update_fat32_fsinfo_sector_retn:
1844 0000C7A6 C3                  <1>     retn
1845                              <1>
1846                              <1> get_fat32_fsinfo_sector_parms:
1847                              <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
1848                              <1>     ; 15/10/2016
1849                              <1>     ; 23/03/2016
1850                              <1>     ; 01/03/2016
1851                              <1>     ; 29/02/2016 (TRDOS 386 = TRDOS v2.0)
1852                              <1>     ; INPUT ->
1853                              <1>     ;       ESI = Logical dos drive description table address
1854                              <1>     ; OUTPUT ->
1855                              <1>     ;       ESI = Logical dos drive description table address
1856                              <1>     ;       EBX = FSINFO sector buffer address (DOSBootSectorBuff)
1857                              <1>     ;       CF = 0 -> OK..
1858                              <1>     ;          EAX = FsInfo sector address
1859                              <1>     ;          ECX = Free cluster count
1860                              <1>     ;          EDX = First free cluster
1861                              <1>     ;       CF = 1 -> Error code in AL (EAX)
1862                              <1>     ;          EBX = 0
1863                              <1>     ;
1864                              <1>     ;       [CFS_FAT32FSINFOSEC] = FAT32 FSINFO sector address
1865                              <1>     ;
1866                              <1>     ; (Modified registers: EAX, EBX, ECX, EDX)
1867                              <1>
1868 0000C7A7 0FB74636            <1>     movzx   eax, word [esi+LD_BPB+FAT32_FSInfoSec]
1869 0000C7AB 03466C              <1>     add     eax, [esi+LD_StartSector]
1870 0000C7AE A3[04810100]        <1>     mov     [CFS_FAT32FSINFOSEC], eax
1871                              <1>
1872 0000C7B3 BB[527C0100]        <1>         mov     ebx, DOSBootSectorBuff
1873                              <1>     ;mov    ecx, 1
1874                              <1>     ; 25/07/2022
1875 0000C7B8 29C9                <1>     sub     ecx, ecx
1876 0000C7BA FEC1                <1>     inc     cl
1877                              <1>     ; ecx = 1
1878 0000C7BC E801520000          <1>     call    disk_read
1879 0000C7C1 722F                <1>     jc      short loc_read_FAT32_fsinfo_sec_err
1880                              <1>
1881 0000C7C3 BB[527C0100]        <1>     mov     ebx, DOSBootSectorBuff
1882                              <1>
1883 0000C7C8 813B52526141        <1>     cmp     dword [ebx], 41615252h
1884 0000C7CE 751E                <1>     jne     short loc_read_FAT32_fsinfo_sec_stc
1885                              <1>
1886 0000C7D0 81BBE4010000727241- <1>     cmp     dword [ebx+484], 61417272h
1886 0000C7D9 61                  <1>
1887 0000C7DA 7512                <1>     jne     short loc_read_FAT32_fsinfo_sec_stc
1888                              <1>
1889 0000C7DC A1[04810100]        <1>     mov     eax, [CFS_FAT32FSINFOSEC]
1890 0000C7E1 8B8BE8010000        <1>     mov     ecx, [ebx+488] ; free cluster count
1891 0000C7E7 8B93EC010000        <1>     mov     edx, [ebx+492] ; first (next) free cluster
1892                              <1>
1893 0000C7ED C3                  <1>     retn
1894                              <1>
1895                              <1> loc_read_FAT32_fsinfo_sec_stc:
1896                              <1>     ; 15/10/2016 (0Bh -> 28)
1897                              <1>     ;mov    eax, 28 ; Invalid format!
1898                              <1>     ; 25/07/2022
1899 0000C7EE B31C                <1>     mov     bl, 28
1900 0000C7F0 EB02                <1>     jmp     short loc_read_FAT32_fsinfo_sec_stc_retn
```

```
                                <1>
                                <1> loc_read_FAT32_fsinfo_sec_err:
                                <1>     ; 15/10/2016 (15h -> 17)
                                <1>     ; 23/03/2016 (15h)
                                <1>     ;mov    eax, 17 ; Drive not ready or read error
                                <1>     ; 25/07/2022
0000C7F2 B311                   <1>     mov     bl, 17
                                <1> loc_read_FAT32_fsinfo_sec_stc_retn:
                                <1>     ; 25/07/2022
0000C7F4 29C0                   <1>     sub     eax, eax
0000C7F6 88D8                   <1>     mov     al, bl ; error code
                                <1>     ; eax = error code
0000C7F8 29DB                   <1>     sub     ebx, ebx ; 0
0000C7FA F9                     <1>     stc
0000C7FB C3                     <1>     retn
                                <1>
                                <1> add_new_cluster:
                                <1>     ; 30/08/2024
                                <1>     ; 27/08/2024
                                <1>     ; 25/08/2024 - TRDOS 386 v2.0.9
                                <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <1>     ; 15/10/2016
                                <1>     ; 16/05/2016
                                <1>     ; 18/03/2016, 24/03/2016
                                <1>     ; 11/03/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>     ; 30/07/2011 (DRV_FAT.ASM)
                                <1>     ; 11/09/2010
                                <1>     ; INPUT ->
                                <1>     ;       ESI = Logical dos drv desc. table address
                                <1>     ;       EAX = Last cluster
                                <1>     ; OUTPUT ->
                                <1>     ;       ESI = Logical dos drv desc. table address
                                <1>     ;       EAX = New Last cluster (next cluster)
                                <1>     ;       cf = 1 -> error code in EAX (AL)
                                <1>     ;       cf = 1 -> EBX = sectors per cluster
                                <1>     ;       ECX = Free sectors
                                <1>     ;;;     25/07/2022
                                <1>     ;       (EBX = sectors per cluster -not used-)
                                <1>     ;       EDX = 0 (if cf = 0)
                                <1>     ; NOTE:
                                <1>     ; This procedure does not update lm date&time !
                                <1>     ;    ; 30/08/2024
                                <1>     ; and doesn't update 1st clust and file size fields !
                                <1>     ;
                                <1>     ; (Modified registers: EAX, EBX, ECX, EDX, EDI)
                                <1>
0000C7FC A3[34820100]           <1>     mov     [FAT_anc_LCluster], eax
                                <1>
0000C801 E885F9FFFF             <1>     call    get_first_free_cluster
0000C806 7206                   <1>     jc      short loc_add_new_cluster_retn
                                <1>     ; EAX >= 2 and EAX < FFFFFFFFh is valid
                                <1>
                                <1>     ;mov    edx, eax
                                <1>     ;
                                <1>     ;inc    edx
                                <1>     ;;jnz   short loc_add_new_cluster_check_ffc_eax
                                <1>     ;jnz    short loc_add_new_cluster_save_fcc
                                <1>
                                <1>     ; 27/08/2024
0000C808 40                     <1>     inc     eax ; (*)
0000C809 750D                   <1>     jnz     short loc_add_new_cluster_save_fcc
                                <1>
                                <1> loc_add_new_cluster_no_disk_space_retn:
                                <1>     ;mov    eax, 27h ; MSDOS err => insufficient disk space
                                <1>     ; 27/08/2024
                                <1>     ; eax = 0
                                <1>     ; 25/07/2022
                                <1>     ;xor    eax, eax
0000C80B B027                   <1>     mov     al, 27h
                                <1> loc_add_new_cluster_stc_retn:
0000C80D F9                     <1>     stc
                                <1> loc_add_new_cluster_retn:
                                <1>     ; 25/07/2022
                                <1>     ;movzx  ebx, byte [esi+LD_BPB+SecPerClust]
0000C80E 8B4E74                 <1>     mov     ecx, [esi+LD_FreeSectors]
                                <1>     ;xor    edx, edx
                                <1>     ;stc
0000C811 C3                     <1>     retn
                                <1>
                                <1> loc_anc_invalid_format_stc_retn:
                                <1>     ; 27/08/2024
                                <1>     ;stc
                                <1> loc_add_new_cluster_invalid_format_retn:
                                <1>     ; 15/10/2016 (0Bh -> 28)
                                <1>     ;mov    eax, 28 ; Invalid format
                                <1>     ;jmp    short loc_add_new_cluster_retn
                                <1>     ; 25/07/2022
0000C812 29C0                   <1>     sub     eax, eax
0000C814 B01C                   <1>     mov     al, 28
0000C816 EBF5                   <1>     jmp     short loc_add_new_cluster_stc_retn
                                <1>
                                <1> ;loc_add_new_cluster_check_ffc_eax:
                                <1> ;   cmp     eax, 2
                                <1> ;   jb      short loc_add_new_cluster_invalid_format_retn
                                <1>
                                <1> loc_add_new_cluster_save_fcc:
                                <1>     ;;;
                                <1>     ; 27/08/2024
0000C818 48                     <1>     dec     eax ; (*)
                                <1>     ;;;
0000C819 A3[38820100]           <1>     mov     [FAT_anc_FFCluster], eax
                                <1>
                                <1> ; 27/08/2024 (TRDOS 386 v2.0.9)
                                <1> %if 0
                                <1>     sub     eax, 2
                                <1>     movzx   ebx, byte [esi+LD_BPB+SecPerClust]
                                <1>     mul     ebx
                                <1>     or      edx, edx
                                <1>     jnz     short loc_anc_invalid_format_stc_retn
                                <1>
                                <1> loc_add_new_cluster_allocate_cluster:
                                <1>     ; 18/03/2016
                                <1>     xchg    edx, eax ; eax = 0
                                <1>     ; 16/05/2016
                                <1>     ;cmp    [ClusterBuffer_Valid], al ; 0
                                <1>     ;jna    short loc_anc_clear_cluster_buffer
                                <1>     ;; 'copy' command,
                                <1>     ;; writing destination file clust after reading source file clust
                                <1>     ;mov    [ClusterBuffer_Valid], al ; 0 ; reset
                                <1>     ;jmp    short loc_add_new_cluster_write_nc_to_disk
                                <1>
                                <1> loc_anc_clear_cluster_buffer:
                                <1>     ; 11/03/2016
                                <1>     ; Clear buffer
```

```
2025                            <1>     mov     edi, Cluster_Buffer ; 70000h (for current TRDOS 386 version)
2026                            <1>     mov     ecx, ebx ; sector count
2027                            <1>     shl     ecx, 7 ; 1 sector = 512 bytes -> 128 double words
2028                            <1>     ;xor    eax, eax ; 0
2029                            <1>     rep     stosd
2030                            <1>
2031                            <1> loc_add_new_cluster_write_nc_to_disk:
2032                            <1>     ; 11/03/2016
2033                            <1>     ;xchg   eax, edx ; edx = 0, eax = sector offset
2034                            <1>     mov     eax, edx
2035                            <1>         add     eax, [esi+LD_DATABegin]
2036                            <1>     jc      short loc_add_new_cluster_invalid_format_retn
2037                            <1>
2038                            <1>     mov     ecx, ebx ; ECX = sectors per cluster (<256)
2039                            <1>     mov     ebx, Cluster_Buffer
2040                            <1>     call    disk_write
2041                            <1>     jnc     short loc_add_new_cluster_update_fat_nlc
2042                            <1>
2043                            <1>     ; 15/10/2016 (1Dh -> 18)
2044                            <1>     ;mov    eax, 18 ; Write Error
2045                            <1>     ; 25/07/2022
2046                            <1>     xor     eax, eax
2047                            <1>     mov     al, 18
2048                            <1>     jmp     short loc_add_new_cluster_stc_retn
2049                            <1>
2050                            <1> loc_add_new_cluster_update_fat_nlc:
2051                            <1>     mov     eax, [FAT_anc_FFCluster]
2052                            <1> %endif
2053                            <1>     ; 30/08/2024
2054                            <1>     ; eax = [FAT_anc_FFCluster] ; first free cluster
2055 0000C81E 31C9               <1>     xor     ecx, ecx ; 0
2056 0000C820 890D[5E7E0100]    <1>     mov     [FAT_ClusterCounter], ecx ; 0 ; reset
2057 0000C826 49                <1>     dec     ecx ; -1 ; 0FFFFFFFFh
2058 0000C827 850D[34820100]    <1>     test    [FAT_anc_LCluster], ecx       ; 0 ?
2059 0000C82D 7416              <1>     jz      short loc_add_new_cluster_update_fat_fc ; yes
2060                            <1>
2061                            <1>     ; 27/08/2024
2062                            <1>     ; eax = (first free) cluster to be added as last cluster
2063                            <1>     ;;;
2064                            <1>     ;xor    ecx, ecx
2065                            <1>
2066                            <1> loc_add_new_cluster_update_fat_nlc: ; 30/08/2024
2067                            <1>     ;mov    [FAT_ClusterCounter], ecx ; 0 ; reset
2068                            <1>     ;dec    ecx ; 0FFFFFFFFh ; last cluster
2069 0000C82F E82CFAFFFF          <1>     call    update_cluster
2070 0000C834 7304              <1>     jnc     short loc_add_new_cluster_update_fat_plc
2071 0000C836 09C0              <1>     or      eax, eax ; EAX = 0 -> cluster value is 0 or eocc
2072 0000C838 75D3              <1>     jnz     short loc_add_new_cluster_stc_retn
2073                            <1>
2074                            <1> loc_add_new_cluster_update_fat_plc:
2075 0000C83A A1[34820100]      <1>     mov     eax, [FAT_anc_LCluster]
2076 0000C83F 8B0D[38820100]    <1>     mov     ecx, [FAT_anc_FFCluster]
2077                            <1> loc_add_new_cluster_update_fat_fc: ; 30/08/2024
2078 0000C845 E816FAFFFF        <1>     call    update_cluster
2079 0000C84A 7311              <1>     jnc     short loc_add_new_cluster_save_fat_buffer
2080                            <1>
2081 0000C84C 09C0              <1>     or      eax, eax ; EAX = 0 -> cluster value is 0 or eocc
2082 0000C84E 740D              <1>     jz      short loc_add_new_cluster_save_fat_buffer
2083                            <1>
2084                            <1> loc_anc_save_fat_buffer_err_retn:
2085                            <1>     ;cmp    byte [FAT_ClusterCounter], 1
2086                            <1>     ;jb     short loc_add_new_cluster_retn
2087                            <1>
2088 0000C850 66BB00FF          <1>     mov     bx, 0FF00h ; recalculate free space (BL = 0)
2089                            <1>                        ; (BH = FFh -> Use ESI as Drv Param. Tbl.)
2090 0000C854 50                <1>     push    eax
2091 0000C855 E807FDFFFF        <1>     call    calculate_fat_freespace
2092 0000C85A 58                <1>     pop     eax
2093 0000C85B EBB0              <1>         jmp     loc_add_new_cluster_stc_retn
2094                            <1>
2095                            <1> loc_add_new_cluster_save_fat_buffer:
2096                            <1>     ;cmp    byte [FAT_BuffValidData], 2
2097                            <1>     ;jne    short loc_add_new_cluster_calc_FAT_freespace
2098                            <1>     ;Byte [FAT_BuffValidData] = 2
2099 0000C85D E872FCFFFF        <1>     call    save_fat_buffer
2100 0000C862 72EC              <1>     jc      short loc_anc_save_fat_buffer_err_retn
2101                            <1>
2102                            <1> loc_add_new_cluster_calc_FAT_freespace:
2103                            <1>     ;mov    eax, 1 ; Only one Cluster
2104 0000C864 A1[5E7E0100]      <1>     mov     eax, [FAT_ClusterCounter]
2105 0000C869 66BB01FF          <1>     mov     bx, 0FF01h ; BH = FFh -> ESI -> Dos drv desc. table
2106                            <1>                        ; BL = 1 -> add cluster(s)
2107                            <1>     ;mov    bl, 01h ; 27/08/2024
2108                            <1>     ; NOTE: EAX value will be added to Free Cluster Count
2109                            <1>     ; (Free Cluster Count is decreased when EAX value is negative)
2110 0000C86D E8EFFCFFFF        <1>     call    calculate_fat_freespace
2111                            <1>     ; ECX = 0 -> no error, ECX > 0 -> error or invalid return
2112 0000C872 21C9              <1>     and     ecx, ecx ; ECX = 0 -> valid free sector count
2113 0000C874 7409              <1>     jz      short loc_add_new_cluster_return_cluster_number
2114                            <1>
2115                            <1> loc_add_new_cluster_recalc_FAT_freespace:
2116 0000C876 66BB00FF          <1>     mov     bx, 0FF00h  ; recalculate free space
2117 0000C87A E8E2FCFFFF        <1>         call    calculate_fat_freespace
2118                            <1>     ; cf = 0
2119                            <1> loc_add_new_cluster_return_cluster_number:
2120 0000C87F 89C1              <1>     mov     ecx, eax ; Free sector count
2121 0000C881 A1[38820100]      <1>     mov     eax, [FAT_anc_FFCluster]
2122                            <1>     ;mov    edi, Cluster_Buffer
2123                            <1>     ; 25/07/2022 (EBX is not used by callers of this sprocedure)
2124                            <1>     ;movzx  ebx, byte [esi+LD_BPB+SecPerClust]
2125 0000C886 31D2              <1>     xor     edx, edx ; 0
2126 0000C888 C3                <1>         retn
2127                            <1>
2128                            <1> write_cluster:
2129                            <1>     ; 19/12/2025 - TRDOS 386 v2.0.10
2130                            <1>     ; 31/08/2024 - TRDOS 386 v2.0.9
2131                            <1>     ; 15/10/2016
2132                            <1>     ; 21/03/2016 (TRDOS 386 = TRDOS v2.0)
2133                            <1>     ;
2134                            <1>     ; INPUT ->
2135                            <1>     ;       EAX = Cluster Number (Sector index for SINGLIX FS)
2136                            <1>     ;       ESI = Logical DOS Drive Description Table address
2137                            <1>     ;       EBX = Cluster (File R/W) Buffer address (max. 64KB)
2138                            <1>     ;       Only for SINGLIX FS:
2139                            <1>     ;       EDX = File Number (The 1st FDT address)
2140                            <1>     ; OUTPUT ->
2141                            <1>     ;       cf = 1 -> Cluster can not be written onto disk
2142                            <1>     ;           EAX > 0 -> Error number
2143                            <1>     ;       cf = 0 -> Cluster has been written successfully
2144                            <1>     ;
2145                            <1>     ; (Modified registers: EAX, ECX, EBX, EDX)
2146                            <1>
2147 0000C889 0FB64E13          <1>     movzx   ecx, byte [esi+LD_BPB+BPB_SecPerClust]
2148                            <1>     ; CL = 1 = [esi+LD_FS_Reserved2] ; SectPerClust for Singlix FS
```

```
                                <1>
                                <1> write_file_sectors: ; 16/03/2016
                                <1>
                                <1> ; 19/12/2025
                                <1> %if 0
                                <1>     cmp     byte [esi+LD_FATType], 0
                                <1>     jna     short write_fs_cluster
                                <1> %endif
                                <1>
                                <1> write_fat_file_sectors:
                                <1>     ; 31/08/2024
                                <1>     ; ecx = sector count (may be different than sectors per cluster)
0000C88D 83E802                 <1>     sub     eax, 2 ; Beginning cluster number is always 2
0000C890 0FB65613               <1>     movzx   edx, byte [esi+LD_BPB+BPB_SecPerClust] ; 18/03/2016
0000C894 F7E2                   <1>     mul     edx
0000C896 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>
0000C899 E815510000             <1>     call    disk_write
0000C89E 7306                   <1>     jnc     short wclust_retn
                                <1>
                                <1>     ; 15/10/2016 (1Dh -> 18)
0000C8A0 B812000000             <1>     mov     eax, 18 ; Drive not ready or write error !
0000C8A5 C3                     <1>     retn
                                <1>
                                <1> wclust_retn:
0000C8A6 29C0                   <1>     sub     eax, eax ; 0
0000C8A8 C3                     <1>     retn
                                <1>
                                <1> ; 19/12/2025
                                <1> %if 0
                                <1>
                                <1> write_fs_cluster:
                                <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <1>     ; 21/03/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>     ; Singlix FS
                                <1>
                                <1>     ; EAX = Cluster number is sector index number of the file (eax)
                                <1>
                                <1>     ; EDX = File number is the first File Descriptor Table address
                                <1>     ;       of the file. (Absolute address of the FDT).
                                <1>
                                <1>     ; eax = sector index (0 for the first sector)
                                <1>     ; edx = FDT0 address
                                <1>             ; 64 KB buffer = 128 sectors (limit)
                                <1>     ;mov    ecx, 128 ; maximum count of sectors (before eof)
                                <1>     ; 25/07/2022
                                <1>     sub     ecx, ecx
                                <1>     mov     cl, 128
                                <1>     ;call   write_fs_sectors
                                <1>     ;retn
                                <1>     ;jmp    short write_fs_sectors
                                <1>
                                <1> write_fs_sectors:
                                <1>     ; 21/03/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>     stc
                                <1>     retn
                                <1>
                                <1> %endif
                                <1>
                                <1> get_cluster_by_index:
                                <1>     ; 19/12/2025 (TRDOS 386 v2.0.10)
                                <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <1>     ; 29/04/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>     ; INPUT ->
                                <1>     ;       EAX = Beginning cluster
                                <1>     ;       EDX = Sector index in disk/file section
                                <1>     ;             (Only for SINGLIX file system!)
                                <1>     ;       ECX = Cluster sequence number after the beginning cluster
                                <1>     ;       ESI = Logical DOS Drive Description Table address
                                <1>     ; OUTPUT ->
                                <1>     ;       EAX = Cluster number
                                <1>     ;       cf = 1 -> Error code in AL (EAX)
                                <1>     ;
                                <1>     ;(Modified registers: EAX, ECX, EBX, EDX)
                                <1>
                                <1> ; 19/12/2025
                                <1> %if 0
                                <1>     cmp     byte [esi+LD_FATType], 1
                                <1>        jb      short get_fs_section_by_index
                                <1> %endif
0000C8A9 3B4E78                 <1>     cmp     ecx, [esi+LD_Clusters]
0000C8AC 7206                   <1>     jb      short gcbi_1
                                <1> gcbi_0:
                                <1>     ;stc
                                <1>     ;mov    eax, 23h ; Cluster not available !
                                <1>                      ; MSDOS error code: FCB unavailable
                                <1>     ; 25/07/2022
0000C8AE 29C0                   <1>     sub     eax, eax
                                <1> gcbi_4:
0000C8B0 B023                   <1>     mov     al, 23h
0000C8B2 F9                     <1>     stc
0000C8B3 C3                     <1>     retn
                                <1> gcbi_1:
0000C8B4 51                     <1>     push    ecx
0000C8B5 E8BFF6FFFF             <1>     call    get_next_cluster
0000C8BA 59                     <1>     pop     ecx
0000C8BB 7203                   <1>     jc      short gcbi_3
0000C8BD E2F5                   <1>     loop    gcbi_1
                                <1> gcbi_2:
0000C8BF C3                     <1>     retn
                                <1> gcbi_3:
0000C8C0 09C0                   <1>     or      eax, eax
                                <1>     ;jz     short gcbi_0
                                <1>     ; 25/07/2022
0000C8C2 74EC                   <1>     jz      short gcbi_4
0000C8C4 F5                     <1>     cmc     ; stc
0000C8C5 C3                     <1>     retn
                                <1>
                                <1> ; 19/12/2025
                                <1> %if 0
                                <1>
                                <1> get_fs_section_by_index:
                                <1>     ; 29/04/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>     ; INPUT ->
                                <1>     ;       EAX = Beginning FDT number/address
                                <1>     ;       EDX = Sector index in disk/file section
                                <1>     ;       ECX = Sector sequence number after the beginning FDT
                                <1>     ;       ESI = Logical DOS Drive Description Table address
```

```
2273                                <1>     ; OUTPUT ->
2274                                <1>     ;       EAX = FDT number/address
2275                                <1>     ;       EDX = Sector index of the section (0,1,2,3,4...)
2276                                <1>     ;       cf = 1 -> Error code in AL (EAX)
2277                                <1>     ;
2278                                <1>     ;(Modified registers: EAX, ECX, EBX, EDX)
2279                                <1>     ;
2280                                <1>     mov     eax, 0FFFFFFFFh
2281                                <1>     retn
2282                                <1>
2283                                <1> get_last_section:
2284                                <1>     ; 22/10/2016 (TRDOS 386 = TRDOS v2.0)
2285                                <1>     ; INPUT ->
2286                                <1>     ;       EAX = (The 1st) FDT number/address
2287                                <1>     ;       ESI = Logical DOS Drive Description Table address
2288                                <1>     ; OUTPUT ->
2289                                <1>     ;       EAX = FDT number/address of the last section
2290                                <1>     ;       EDX = Last sector of the section (0,1,2,3,4...)
2291                                <1>     ;       [glc_index] = sector index number of the last sector
2292                                <1>     ;                     (for file, not for the last section)
2293                                <1>     ;
2294                                <1>     ;       cf = 1 -> Error code in AL (EAX)
2295                                <1>     ;
2296                                <1>     ;(Modified registers: EAX, ECX, EBX, EDX)
2297                                <1>     ;
2298                                <1>     mov     eax, 0
2299                                <1>     mov     edx, 0
2300                                <1>     retn
2301                                <1>
2302                                <1> %endif
3437                                    %include 'trdosk6.s' ; 24/01/2016
   1                                <1> ; ****************************************************************************
   2                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.10) - MAIN PROGRAM : trdosk6.s
   3                                <1> ; ----------------------------------------------------------------------------
   4                                <1> ; Last Update: 27/12/2025  (Previous: 27/09/2024, v2.0.9)
   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> ; Ref: Retro UNIX 386 v1.2 Kernel (v0.2.2.3) - ux.s - 15/07/2022
  19                                <1>
  20                                <1> sysent: ; < enter to system call >
  21                                <1>     ; 23/07/2022 - TRDOS 386 Kernel v2.0.5
  22                                <1>     ; 17/03/2017
  23                                <1>     ; 03/03/2017
  24                                <1>     ; 19/02/2017
  25                                <1>     ; 13/01/2017
  26                                <1>     ; 06/06/2016
  27                                <1>     ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
  28                                <1>     ; 16/04/2015 - 19/10/2015 (Retro UNIX 386 v1)
  29                                <1>     ; 10/04/2013 - 18/01/2014 (Retro UNIX 8086 v1)
  30                                <1>     ;
  31                                <1>     ; 'unkni' or 'sysent' is sytem entry from various traps.
  32                                <1>     ; The trap type is determined and an indirect jump is made to
  33                                <1>     ; the appropriate system call handler. If there is a trap inside
  34                                <1>     ; the system a jump to panic is made. All user registers are saved
  35                                <1>     ; and u.sp points to the end of the users stack. The sys (trap)
  36                                <1>     ; instructor is decoded to get the the system code part (see
  37                                <1>     ; trap instruction in the PDP-11 handbook) and from this
  38                                <1>     ; the indirect jump address is calculated. If a bad system call is
  39                                <1>     ; made, i.e., the limits of the jump table are exceeded, 'badsys'
  40                                <1>     ; is called. If the call is legitimate control passes to the
  41                                <1>     ; appropriate system routine.
  42                                <1>     ;
  43                                <1>     ; Calling sequence:
  44                                <1>     ;       Through a trap caused by any sys call outside the system.
  45                                <1>     ; Arguments:
  46                                <1>     ;       Arguments of particular system call.
  47                                <1>     ; ....................................................
  48                                <1>     ;
  49                                <1>     ; Retro UNIX 8086 v1 modification:
  50                                <1>     ;       System call number is in EAX register.
  51                                <1>     ;
  52                                <1>     ;       Other parameters are in EDX, EBX, ECX, ESI, EDI, EBP
  53                                <1>     ;       registers depending of function details.
  54                                <1>     ;
  55                                <1>     ; 16/04/2015
  56 0000C8C6 368925[2C8E0100]      <1>       mov     [ss:u.sp], esp ; Kernel stack points to return address
  57                                <1>
  58                                <1>     ; save user registers
  59 0000C8CD 1E                    <1>     push    ds
  60 0000C8CE 06                    <1>     push    es
  61 0000C8CF 0FA0                  <1>     push    fs
  62 0000C8D1 0FA8                  <1>     push    gs
  63 0000C8D3 60                    <1>     pushad  ; eax, ecx, edx, ebx, esp -before pushad-, ebp, esi, edi
  64                                <1>     ;
  65                                <1>     ; ESPACE = [ss:u.sp] - esp ; 4*12 = 48 ; 17/09/2015 ; 06/06/2016
  66                                <1>     ;       (ESPACE is size of space in kernel stack
  67                                <1>     ;       for saving/restoring user registers.)
  68                                <1>     ;
  69 0000C8D4 50                    <1>     push    eax ; 01/07/2015
  70 0000C8D5 66B81000              <1>     mov     ax, KDATA
  71 0000C8D9 8ED8                  <1>         mov     ds, ax
  72 0000C8DB 8EC0                  <1>         mov     es, ax
  73 0000C8DD 8EE0                  <1>         mov     fs, ax
  74 0000C8DF 8EE8                  <1>         mov     gs, ax
  75 0000C8E1 A1[80760100]          <1>     mov     eax, [k_page_dir]
  76 0000C8E6 0F22D8                <1>     mov     cr3, eax
  77 0000C8E9 58                    <1>     pop     eax ; 01/07/2015
  78                                <1>     ; 19/10/2015
  79 0000C8EA FC                    <1>     cld
  80                                <1>     ;
  81 0000C8EB FE05[288E0100]        <1>     inc     byte [sysflg]
  82                                <1>             ; incb sysflg / indicate a system routine is in progress
  83 0000C8F1 FB                    <1>         sti         ; 18/01/2014
  84                                <1>     ;jnz    panic ; 24/05/2013
  85                                <1>             ; beq 1f
  86                                <1>             ; jmp panic ; / called if trap inside system
  87                                <1>     ; 23/07/2022
  88 0000C8F2 7405                  <1>     jz      short sysent0
  89 0000C8F4 E910A5FFFF            <1>     jmp     panic
  90                                <1> ;1:
  91                                <1> sysent0:
  92                                <1>     ; 17/03/2017
  93 0000C8F9 80642438FE            <1>     and     byte [esp+ESPACE+8], ~1 ; clear carry flag
```

```
                                <1>
                                <1>     ; 16/04/2015
0000C8FE A3[348E0100]           <1>     mov     [u.r0], eax
0000C903 8925[308E0100]         <1>     mov     [u.usp], esp ; kernel stack points to user's registers
                                <1>
                                <1>     ; 13/01/2017 (TRDOS 386 Feaure only !)
0000C909 803D[AC8E0100]00       <1>     cmp     byte [u.t_lock], 0 ; timer interrupt lock ?
                                <1>     ;ja     sysrele          ; yes, sys release only !!!
                                <1>     ; 23/07/2022
0000C910 7605                   <1>     jna     short sysent1
0000C912 E902020000             <1>     jmp     sysrele
                                <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
                                <1>             ; sub $sys,r0 / get xxx code
                                <1> sysent1:
0000C917 C1E002                 <1>     shl     eax, 2
                                <1>             ; asl r0 / multiply by 2 to jump indirect in bytes
0000C91A 3DBC000000             <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
0000C91F F5                     <1>     cmc
0000C920 9C                     <1>     pushf
0000C921 50                     <1>     push    eax
0000C922 8B2D[2C8E0100]         <1>     mov     ebp, [u.sp] ; Kernel stack at the beginning of sys call
0000C928 B0FE                   <1>     mov     al, 0FEh ; 11111110b
0000C92A 1400                   <1>     adc     al, 0 ; al = al + cf
0000C92C 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
0000C92F 5D                     <1>     pop     ebp ; eax
0000C930 9D                     <1>     popf
0000C931 720B                   <1>         jc      short badsys ; 23/07/2022
0000C933 A1[348E0100]           <1>     mov     eax, [u.r0]
                                <1>     ; system call registers: EAX, EDX, ECX, EBX, ESI, EDI
0000C938 FFA5[8BC90000]         <1>     jmp     dword [ebp+syscalls]
                                <1>             ; jmp *1f(r0) / jump indirect thru table of addresses
                                <1>                         ; / to proper system routine.
                                <1>
                                <1>     ; 20/08/2024 (exit code)
                                <1>     ; 30/07/2022
                                <1>     ; 23/07/2022
                                <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>     ;
0000C93E FE05[848E0100]         <1>     inc     byte [u.bsys]
                                <1>     ;
0000C944 8B1D[2C8E0100]         <1>     mov     ebx, [u.sp] ; esp at the beginning of 'sysent'
0000C94A 8B03                   <1>     mov     eax, [ebx] ; EIP (return address, not 'INT 30h' address)
                                <1>     ;sub    eax, 2 ; CDh, ##h
                                <1>     ; 30/07/2022
0000C94C 48                     <1>     dec     eax
0000C94D 48                     <1>     dec     eax
0000C94E E8A878FFFF             <1>     call    dwordtohex
0000C953 8915[73350100]         <1>     mov     [eip_str], edx
0000C959 A3[77350100]           <1>     mov     [eip_str+4], eax
0000C95E A1[348E0100]           <1>     mov     eax, [u.r0]
0000C963 E89378FFFF             <1>     call    dwordtohex
0000C968 8915[62350100]         <1>     mov     [eax_str], edx
0000C96E A3[66350100]           <1>     mov     [eax_str+4], eax
                                <1>
0000C973 66C705[57350100]34-    <1>     mov     word [int_num_str], SYSCALL_INT_NUM ; 25/12/2016
0000C97B 30                     <1>
                                <1>
0000C97C BE[29350100]           <1>     mov     esi, ifc_msg ; "invalid function call !" msg (trdosk9.s)
                                <1>
0000C981 E824A4FFFF             <1>     call    print_msg
                                <1>
                                <1> ;; 20/08/2024 - temporary
                                <1> ;   mov     ebx, 07h
                                <1> ;   mov     ah, 0Eh
                                <1> ;p_fc_msg:
                                <1> ;   lodsb
                                <1> ;   and     al, al
                                <1> ;   jz      short p_fc_ms_ok
                                <1> ;   call    _int10h
                                <1> ;   jmp     short p_fc_msg
                                <1> ;p_fc_ms_ok:
                                <1>
                                <1>     ;jmp    sysexit
                                <1>
                                <1>     ; 20/08/2024 (exit code)
                                <1>     ;mov    bl, 0FFh ; -1
                                <1>     ;jmp    sysexit
                                <1>     ;
0000C986 E92C020000             <1>     jmp     sysexit_0FFh
                                <1>
                                <1> syscalls: ; 1:
                                <1>     ; 20/08/2024 - TRDOS 386 v2.0.9
                                <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)
0000C98B [360A0100]             <1>     dd sysver       ; 0 ; Get TRDOS 386 version number (v2.0)
0000C98F [BCCB0000]             <1>     dd sysexit      ; 1
0000C993 [E5CD0000]             <1>     dd sysfork      ; 2
0000C997 [B2D10000]             <1>     dd sysread      ; 3
0000C99B [D1D10000]             <1>     dd syswrite     ; 4
0000C99F [CFCF0000]             <1>     dd sysopen      ; 5
0000C9A3 [88D10000]             <1>     dd sysclose     ; 6
0000C9A7 [49CD0000]             <1>     dd syswait      ; 7
0000C9AB [F2CE0000]             <1>     dd syscreat     ; 8
0000C9AF [22170100]             <1>     dd sysrename    ; 9  ; TRDOS 386, Rename File (31/12/2017)
0000C9B3 [CF120100]             <1>     dd sysdelete    ; 10 ; TRDOS 386, Delete File (29/12/2017)
0000C9B7 [7FFE0000]             <1>     dd sysexec      ; 11
0000C9BB [FF130100]             <1>     dd syschdir     ; 12
0000C9BF [8C150100]             <1>     dd systime      ; 13 ; TRDOS 386, Get Sys Date&Time (30/12/2017)
0000C9C3 [4BD10000]             <1>     dd sysmkdir     ; 14
0000C9C7 [33140100]             <1>     dd syschmod     ; 15 ; TRDOS 386, Change Attributes (30/12/2017)
0000C9CB [36130100]             <1>     dd sysrmdir     ; 16 ; TRDOS 386, Remove Directory (29/12/2017)
0000C9CF [07030100]             <1>     dd sysbreak     ; 17
0000C9D3 [E1140100]             <1>     dd sysdrive     ; 18 ; TRDOS 386, Get/Set Current Drv (30/12/2017)
0000C9D7 [76030100]             <1>     dd sysseek      ; 19
0000C9DB [8D030100]             <1>     dd systell      ; 20
0000C9DF [2D180100]             <1>     dd sysmem       ; 21 ; TRDOS 386, Get Total&Free Mem (31/12/2017)
0000C9E3 [63180100]             <1>     dd sysprompt    ; 22 ; TRDOS 386, Change Cmd Prompt (31/12/2017)
0000C9E7 [A4180100]             <1>     dd syspath      ; 23 ; TRDOS 386, Get/Set Run Path (31/12/2017)
0000C9EB [09190100]             <1>     dd sysenv       ; 24 ; TRDOS 386, Get/Set Env Vars (31/12/2017)
0000C9EF [0D160100]             <1>     dd sysstime     ; 25 ; TRDOS 386, Set Sys Date&Time (30/12/2017)
0000C9F3 [EC030100]             <1>     dd sysquit      ; 26
0000C9F7 [D5030100]             <1>     dd sysintr      ; 27
0000C9FB [30150100]             <1>     dd sysdir       ; 28 ; TRDOS 386, Get Curr Drive&Dir (30/12/2017)
0000C9FF [47CA0000]             <1>     dd sysemt       ; 29
0000CA03 [6B150100]             <1>     dd sysldrvt     ; 30 ; TRDOS 386, Get Logical DOS DDT (30/12/2017)
0000CA07 [CCD30000]             <1>     dd sysvideo     ; 31 ; TRDOS 386 Video Functions (16/05/2016)
0000CA0B [31200100]             <1>     dd sysaudio     ; 32 ; TRDOS 386 Audio Functions (16/05/2016)
0000CA0F [37D20000]             <1>     dd systimer     ; 33 ; TRDOS 386 Timer Functions (18/05/2016)
0000CA13 [08040100]             <1>     dd syssleep     ; 34 ; Retro UNIX 8086 v1 feature only !
                                <1>                             ; 11/06/2014
0000CA17 [48040100]             <1>     dd sysmsg       ; 35 ; Retro UNIX 386 v1 feature only !
                                <1>                             ; 01/07/2015
0000CA1B [63050100]             <1>     dd sysgeterr    ; 36 ; Retro UNIX 386 v1 feature only !
                                <1>                             ; 21/09/2015 - get last error number
0000CA1F [A6120100]             <1>     dd sysfpstat    ; 37 ; TRDOS 386 FPU state option (28/02/2017)
0000CA23 [450A0100]             <1>     dd syspri       ; 38 ; change priority - TRDOS 386 (20/05/2016)
0000CA27 [19CB0000]             <1>     dd sysrele      ; 39 ; TRDOS 386 (19/05/2016) (0 -> 39)
0000CA2B [700B0100]             <1>     dd sysfff       ; 40 ; Find First File - TRDOS 386 (15/10/2016)
0000CA2F [4A0C0100]             <1>     dd sysfnf       ; 41 ; Find Next File - TRDOS 386 (15/10/2016)
0000CA33 [94110100]             <1>     dd sysalloc     ; 42 ; Allocate contiguous memory block/pages
                                <1>                             ; TRDOS 386 (19/02/2017) DMA buff fuctions
0000CA37 [4E120100]             <1>     dd sysdalloc    ; 43 ; Deallocate contiguous memory block/pages
                                <1>                             ; TRDOS 386 (19/02/2017) DMA buff fuctions
0000CA3B [89120100]             <1>     dd syscalbac    ; 44 ; IRQ Callback and Signal Response Byte
                                <1>                             ; service setup - TRDOS 386 (20/02/2017)
                                <1>                             ; 28/08/2017 (20/08/2017)
0000CA3F [12290100]             <1>     dd sysdma       ; 45 ; TRDOS 386 - (ISA) DMA service
0000CA43 [4E2C0100]             <1>     dd sysstdio     ; 46 ; TRDOS 386 v2.0.9 (STDIN/STDOUT functions)
                                <1>
                                <1> end_of_syscalls:
                                <1>
                                <1> sysemt: ; enable (or disable) multi tasking -time sharing-
                                <1>     ;
                                <1>     ; 08/08/2022
                                <1>     ; 23/07/2022 - TRDOS 386 v2.0.5
                                <1>     ; 23/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>     ; 14/05/2015 (Retro UNIX 386 v1)
                                <1>     ; 10/12/2013 - 20/04/2014 (Retro UNIX 8086 v1)
                                <1>     ;
                                <1>     ; Retro UNIX 8086 v1 modification:
                                <1>     ;       'Enable Multi Tasking'  system call instead
                                <1>     ;       of 'Emulator Trap' in original UNIX v1 for PDP-11.
                                <1>     ;
                                <1>     ; Retro UNIX 8086 v1 feature only!
                                <1>     ;       Using purpose: Kernel will start without time-out
                                <1>     ;       (internal clock/timer) functionality.
                                <1>     ;       Then etc/init will enable clock/timer for
                                <1>     ;       multi tasking.
                                <1>     ;
                                <1>     ; INPUT ->
                                <1>     ;       BL = 0 -> disable multi tasking
                                <1>     ;       BL > 1 -> enable multi tasking (time sharing)
                                <1>     ; OUTPUT ->
                                <1>     ;       none
                                <1>     ;
                                <1>     ;  Note: Multi tasking is disabled during system
                                <1>     ;        initialization, it must be enabled by using
                                <1>     ;        this system call. (Otherwise, running proces
                                <1>     ;        will not be changed by another process within
                                <1>     ;        run time sequence/schedule, if running process
                                <1>     ;        will not 'release' itself. Only 'wakeup' procedure
                                <1>     ;        for waiting processes and programmed timer events
                                <1>     ;        for other processes can change running process
                                <1>     ;        while multi tasking is disabled.) ** 23/05/2016 **
                                <1>
0000CA47 803D[868E0100]00       <1>     cmp     byte [u.uid], 0 ; root ?
                                <1>     ;ja     short error
                                <1>     ; 23/07/2022
                                <1>     ;ja     short badsys ; 14/05/2015
                                <1>     ; 08/08/2022
0000CA4E 7605                   <1>     jna     short sysemt_root
0000CA50 E9E9FEFFFF             <1>     jmp     badsys
                                <1> sysemt_root:        ; 08/08/2022
0000CA55 FA                     <1>     cli
0000CA56 881D[12830100]         <1>     mov     [multi_tasking], bl ; 0 to disable, >0 to enable
0000CA5C EB20                   <1>     jmp     sysret
                                <1>
                                <1> error:
                                <1>     ; 18/05/2016
                                <1>     ; 13/05/2016
                                <1>     ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>     ; 16/04/2015 - 17/09/2015 (Retro UNIX 386 v1)
                                <1>     ; 10/04/2013 - 07/08/2013 (Retro UNIX 8086 v1)
                                <1>     ;
                                <1>     ; 'error' merely sets the error bit off the processor status (c-bit)
                                <1>     ; then falls right into the 'sysret', 'sysrele' return sequence.
                                <1>     ;
                                <1>     ; INPUTS -> none
                                <1>     ; OUTPUTS ->
                                <1>     ;       processor status - carry (c) bit is set (means error)
                                <1>     ;
                                <1>     ; 26/05/2013 (Stack pointer must be reset here!
                                <1>     ;             Because, jumps to error procedure
                                <1>     ;             disrupts push-pop nesting balance)
                                <1>     ;
0000CA5E 8B2D[2C8E0100]         <1>     mov     ebp, [u.sp] ; interrupt (system call) return (iretd) address
0000CA64 804D0801               <1>     or      byte [ebp+8], 1  ; set carry bit of flags register
```

```
                                <1>                             ; (system call will return with cf = 1)
                                <1>             ; bis $1,20.(r1) / set c bit in processor status word below
                                <1>                             ; / users stack
                                <1>     ; 17/09/2015
0000CA68 83ED30                 <1>     sub     ebp, ESPACE ; 48 ; total size of stack frame ('sysdefs.inc')
                                <1>                             ; for saving/restoring user registers
                                <1>     ;cmp    ebp, [u.usp]
                                <1>     ;je     short err0
0000CA6B 892D[308E0100]         <1>     mov     [u.usp], ebp
                                <1> ;err0:
                                <1>     ; 01/09/2015
0000CA71 8B25[308E0100]         <1>     mov     esp, [u.usp]       ; Retro Unix 8086 v1 modification!
                                <1>                                ; 10/04/2013
                                <1>                                ; (If an I/O error occurs during disk I/O,
                                <1>                                ; related procedures will jump to 'error'
                                <1>                                ; procedure directly without returning to
                                <1>                                ; the caller procedure. So, stack pointer
                                <1>                                    ; must be restored here.)
                                <1>     ; 13/05/2016
                                <1>     ; NOTE: (The last) error code is in 'u.error', it can be retrieved by
                                <1>     ;       'get last error' system call later.
                                <1>
                                <1>     ; 03/09/2015 - 09/06/2015 - 07/08/2013
0000CA77 C605[9A8E0100]00       <1>     mov     byte [u.kcall], 0 ; namei_r, mkdir_w reset
                                <1>
                                <1> sysret: ; < return from system call>
                                <1>     ; 28/08/2024 - TRDOS 386 Kernel v2.0.9
                                <1>     ; 23/07/2022 - TRDOS 386 Kernel v2.0.5
                                <1>     ; 01/03/2017
                                <1>     ; 28/02/2017
                                <1>     ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>     ; 16/04/2015 - 10/09/2015 (Retro UNIX 386 v1)
                                <1>     ; 10/04/2013 - 23/02/2014 (Retro UNIX 8086 v1)
                                <1>     ;
                                <1>     ; 'sysret' first checks to see if process is about to be
                                <1>     ; terminated (u.bsys). If it is, 'sysexit' is called.
                                <1>     ; If not, following happens:
                                <1>     ;       1) The user's stack pointer is restored.
                                <1>     ;       2) r1=0 and 'iget' is called to see if last mentioned
                                <1>     ;          i-node has been modified. If it has, it is written out
                                <1>     ;          via 'ppoke'.
                                <1>     ;       3) If the super block has been modified, it is written out
                                <1>     ;          via 'ppoke'.
                                <1>     ;       4) If the dismountable file system's super block has been
                                <1>     ;          modified, it is written out to the specified device
                                <1>     ;          via 'ppoke'.
                                <1>     ;       5) A check is made if user's time quantum (uquant) ran out
                                <1>     ;          during his execution. If so, 'tswap' is called to give
                                <1>     ;          another user a chance to run.
                                <1>     ;       6) 'sysret' now goes into 'sysrele'.
                                <1>     ;           (See 'sysrele' for conclusion.)
                                <1>     ;
                                <1>     ; Calling sequence:
                                <1>     ;       jump table or 'br sysret'
                                <1>     ; Arguments:
                                <1>     ;       -
                                <1>     ; ..........................................................
                                <1>     ;
                                <1>     ; ((AX=r1 for 'iget' input))
                                <1>     ;
0000CA7E 31C0                   <1>     xor     eax, eax ; 28/02/2017
                                <1> sysret0: ; 29/07/2015 (eax = 0, jump from sysexec)
0000CA80 FEC0                   <1>     inc     al ; 04/05/2013
0000CA82 3805[848E0100]         <1>     cmp     [u.bsys], al ; 1
                                <1>             ; tstb u.bsys / is a process about to be terminated because
                                <1>         ;jnb        sysexit ; 04/05/2013
                                <1>             ; bne sysexit / of an error? yes, go to sysexit
                                <1>     ; 23/07/2022
0000CA88 7205                   <1>     jb      short sysret1
                                <1>     ;jmp    sysexit
                                <1>     ; 22/08/2024
0000CA8A E92B010000             <1>     jmp     sysexit_@ ; BL = 0FFh ; -1
                                <1> sysret1:    ; 23/07/2022
                                <1>     ;mov    esp, [u.usp] ; 24/05/2013 (that is not needed here)
                                <1>             ; mov u.sp,sp / no point stack to users stack
0000CA8F FEC8                   <1>     dec     al ; mov ax, 0
                                <1>             ; clr r1 / zero r1 to check last mentioned i-node
0000CA91 E81C4F0000             <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:
0000CA96 803D[7C8E0100]00       <1>     cmp     byte [u.quant], 0 ; 16/05/2013
                                <1>             ; tstb uquant / is the time quantum 0?
0000CA9D 7705                   <1>         ja      short swapret
                                <1>             ; bne 1f / no, don't swap it out
                                <1> sysrelease: ; 07/12/2013 (jump from 'clock')
0000CA9F E85B3E0000             <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
0000CAA4 E8024F0000             <1>     call    isintr
                                <1>     ; 20/10/2013
0000CAA9 7405                   <1>     jz      short sysrel1
0000CAAB E8F4000000             <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:
0000CAB0 FA                     <1>     cli     ; 14/10/2015
                                <1> sysrel2:
                                <1>     ; 28/02/2017
                                <1>     ; Check if there is a (delayed) callback for current user/process
0000CAB1 A0[AF8E0100]           <1>     mov     al, [u.irqwait]
0000CAB6 240F                   <1>     and     al, 0Fh ; is there a waiting IRQ callback service ?
0000CAB8 7444                   <1>     jz      short sysrel8 ; no
                                <1>
                                <1>     ; Set return to IRQ callback service and return from the service
0000CABA 0FB6D8                 <1>     movzx   ebx, al
0000CABD 883D[AF8E0100]         <1>     mov     [u.irqwait], bh ; 0 ; reset
0000CAC3 8A9B[80350100]         <1>     mov     bl, [ebx+IRQenum] ; (available) IRQ index +1 (1 to 9)
                                <1>     ; 01/03/2017
0000CAC9 FECB                   <1>     dec     bl ; IRQ index number, 0 to 8
0000CACB 7831                   <1>     js      short sysrel8 ; 0 -> FFh (not in use!?)
                                <1>     ;
0000CACD A0[858E0100]           <1>     mov     al, [u.uno] ; current process (user) number
0000CAD2 3883[60880100]         <1>     cmp     [ebx+IRQ.owner], al
0000CAD8 7524                   <1>     jne     short sysrel8 ; it is not the current user/process !?
0000CADA F683[72880100]01       <1>     test    byte [ebx+IRQ.method], 1 ; callback ?
0000CAE1 741B                   <1>     jz      short sysrel8 ; not a callback method !?
                                <1>
0000CAE3 8B93[84880100]         <1>     mov     edx, [ebx+IRQ.addr] ; IRQ callback service address (virtual)
0000CAE9 C605[B08E0100]01       <1>     mov     byte [u.r_lock], 1 ; IRQ callback service in progress flag
                                <1>
0000CAF0 E8B13E0000             <1>     call    wswap ; save user's registers & status
                                <1>                   ;(for return from IRQ callback service)
                                <1>
0000CAF5 8B2D[2C8E0100]         <1>     mov     ebp, [u.sp]; kernel's stack, points to EIP (user)
0000CAFB 895500                 <1>     mov     [ebp], edx ; IRQ call back service address
                                <1> sysrel8:
0000CAFE FE0D[288E0100]         <1>     dec     byte [sysflg]
                                <1>             ; decb sysflg / turn system flag off
                                <1>
0000CB04 A1[8C8E0100]           <1>     mov     eax, [u.pgdir]
0000CB09 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
0000CB0C 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>     ;
0000CB0D A1[348E0100]           <1>     mov     eax, [u.r0]  ; ((return value in EAX))
0000CB12 0FA9                   <1>     pop     gs
0000CB14 0FA1                   <1>     pop     fs
0000CB16 07                     <1>     pop     es
0000CB17 1F                     <1>     pop     ds
                                <1>     ;or     word [esp+8], 200h ; 22/01/2017 ; force enabling interrupts
0000CB18 CF                     <1>     iretd
                                <1>             ; rti / no, return from interrupt
                                <1>
                                <1> sysrele:
                                <1>     ; 23/07/2022 - TRDOS 386 v2.0.5
                                <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
0000CB19 803D[AC8E0100]00       <1>     cmp     byte [u.t_lock], 0 ; TIMER INT LOCK
                                <1>     ;je     short sysrel0 ; classic (Retro UNIX 386 type) sysrele
0000CB20 7735                   <1>     ja      short sysrel3
                                <1>     ; 27/02/2017
0000CB22 803D[B08E0100]00       <1>     cmp     byte [u.r_lock], 0 ; IRQ callback lock
                                <1>     ;jna    sysrel0 ; classic sysrele ; 24/03/2017
                                <1>     ; 23/07/2022
0000CB29 7705                   <1>     ja      short sysrel9
0000CB2B E966FFFFFF             <1>     jmp     sysrel0
                                <1> sysrel9:    ; 23/07/2022
0000CB30 E859000000             <1>     call    sysrel7
0000CB35 803D[B08E0100]00       <1>     cmp     byte [u.r_lock], 0 ; IRQ callback service lock
0000CB3C 7628                   <1>     jna     short sysrel4
0000CB3E C605[B08E0100]00       <1>     mov     byte [u.r_lock], 0 ; reset
                                <1>     ;mov    byte [u.irqwait], 0 ; reset ; 28/02/2017
0000CB45 A0[B18E0100]           <1>     mov     al, [u.r_mode]
0000CB4A 08C0                   <1>     or      al, al
0000CB4C 7518                   <1>     jnz     short sysrel4
0000CB4E FEC8                   <1>     dec     al
0000CB50 A2[B18E0100]           <1>     mov     [u.r_mode], al ; 0FFh ; not necessary !?
0000CB55 EB32                   <1>     jmp     short sysrel6
                                <1> sysrel3:
                                <1>     ; 27/02/2017
0000CB57 E832000000             <1>     call    sysrel7
                                <1>     ; 14/01/2017
0000CB5C 28C0                   <1>     sub     al, al
0000CB5E 3805[AC8E0100]         <1>     cmp     [u.t_lock], al ; 0 ; TIMER INT LOCK
0000CB64 770E                   <1>     ja      short sysrel5 ; yes
                                <1> sysrel4:
                                <1>     ; 29/01/2017
0000CB66 8B44241C               <1>     mov     eax, [esp+28] ; eax
0000CB6A A3[348E0100]           <1>     mov     [u.r0], eax
0000CB6F E93DFFFFFF             <1>     jmp     sysrel2
                                <1> sysrel5:
0000CB74 A2[AC8E0100]           <1>     mov     [u.t_lock], al ; 0 ; reset
0000CB79 A0[AD8E0100]           <1>     mov     al, [u.t_mode]
0000CB7E 20C0                   <1>     and     al, al
                                <1>     ;jnz    short sysrel2 ; 0FFh ; user mode
0000CB80 75E4                   <1>     jnz     short sysrel4 ; 29/01/2017
0000CB82 FEC8                   <1>     dec     al
0000CB84 A2[AD8E0100]           <1>     mov     [u.t_mode], al ; 0FFh ; not necessary !?
```

```
                            <1> sysrel6:
                            <1>     ; cpu will continue from the interrupted sytem call addr
0000CB89 61                 <1>     popad           ; edi, esi, ebp, esp, ebx, edx, ecx, eax
0000CB8A 83C410             <1>     add     esp, 16 ; pass segment segisters: ds, es, fs, gs
0000CB8D CF                 <1>     iretd           ; eip, cs, eflags
                            <1>
                            <1> sysrel7:
0000CB8E 0FB61D[858E0100]   <1>     movzx   ebx, byte [u.uno] ; current process number
                            <1>     ;shl    bx, 2
                            <1>     ; 23/07/2022
0000CB95 C1E302             <1>     shl     ebx, 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
0000CB98 8B83[748D0100]     <1>     mov     eax, [ebx+p.upage-4] ; UPAGE address
0000CB9E FA                 <1>     cli     ; disable interrupts till 'iretd'
0000CB9F E93A3E0000         <1>     jmp     rswap ; restore process 'u' structure
                            <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
0000CBA4 FB                 <1>     sti
                            <1>     ; 07/12/2013
0000CBA5 66FF05[828E0100]   <1>     inc     word [u.quit]
0000CBAC 7408               <1>     jz      short intrct0 ; FFFFh -> 0
0000CBAE 66FF0D[828E0100]   <1>     dec     word [u.quit]
                            <1>     ; 16/04/2015
0000CBB5 C3                 <1>     retn
                            <1> intrct0:
0000CBB6 58                 <1>     pop     eax ; call intract -> retn
                            <1>     ;
                            <1>     ; 20/08/2024
                            <1>     ;xor    eax, eax
                            <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>     ; 20/08/2024
                            <1>     ; ctrl+break -> exit code = -1
                            <1> sysexit_0FFh:
                            <1>     ; 20/08/2024
0000CBB7 31C0               <1>     xor     eax, eax
0000CBB9 40                 <1>     inc     eax     ; mov eax, 1
                            <1> sysexit_@:  ; 22/08/2024
0000CBBA B3FF               <1>     mov     bl, 0FFh ;  exit code ; -1
                            <1>
                            <1> sysexit: ; <terminate process>
                            <1>     ; 22/08/2024
                            <1>     ; 20/08/2024
                            <1>     ; 18/08/2024 - TRDOS 386 v2.0.9
                            <1>     ; 30/07/2022
                            <1>     ; 23/07/2022 - TRDOS 386 v2.0.5
                            <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
                            <1>     ;dec    ax ; 0
                            <1>     ; 18/08/2024
0000CBBC 48                 <1>     dec     eax
0000CBBD 66A3[808E0100]     <1>     mov     [u.intr], ax ; 0
                            <1>             ; clr u.intr / clear interrupt control word
                            <1>             ; clr r1 / clear r1
                            <1>     ; 18/08/2024 - exit code (in EBX, BL)
0000CBC3 881D[B28E0100]     <1>     mov     [u.exit], bl
                            <1> sysexit_0:
                            <1>     ; 30/07/2022
                            <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
0000CBC9 FA                 <1>     cli     ; disable interrupts
                            <1>     ; 23/01/2017 - reset timer frequency (to 18.2Hz)
0000CBCA B036               <1>     mov     al, 00110110b ; 36h
0000CBCC E643               <1>     out     43h, al
0000CBCE 28C0               <1>     sub     al, al ; 0
0000CBD0 E640               <1>     out     40h, al ; LB
0000CBD2 E640               <1>     out     40h, al ; HB
                            <1>     ;
0000CBD4 0FB61D[858E0100]   <1>     movzx   ebx, byte [u.uno]
                            <1>     ;mov    bl, [u.uno] ; process number of dying process
0000CBDB 3883[B78D0100]     <1>     cmp     byte [ebx+p.timer-1], al ; 0
0000CBE1 7639               <1>     jna     short sysexit_12 ; no timer events for this process
0000CBE3 8883[B78D0100]     <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
0000CBE9 8A0D[13830100]     <1>     mov     cl, [timer_events] ; 14/11/2017
                            <1>     ;cli    ; disable interrupts
0000CBEF B410               <1>     mov     ah, 16 ; number of available timer events
0000CBF1 BE[488F0100]       <1>     mov     esi, timer_set ; beginning address of timer events
                            <1> sysexit_7:
0000CBF6 8A06               <1>     mov     al, [esi] ; process number (of timer event)
0000CBF8 38D8               <1>     cmp     al, bl ; process number comparison
0000CBFA 7411               <1>     je      short sysexit_10
0000CBFC 20C0               <1>     and     al, al
0000CBFE 7404               <1>     jz      short sysexit_9
                            <1> sysexit_8:
0000CC00 FEC9               <1>     dec     cl
0000CC02 7416               <1>     jz      short sysexit_11
                            <1> sysexit_9:
0000CC04 FECC               <1>     dec     ah
0000CC06 7414               <1>     jz      short sysexit_12
0000CC08 83C610             <1>     add     esi, 16
0000CC0B EBE9               <1>     jmp     short sysexit_7
                            <1>
                            <1> sysexit_10:
                            <1>     ;mov    byte [esi], 0
0000CC0D 66C7060000         <1>     mov     word [esi], 0
                            <1>     ;mov    dword [esi+12], 0
                            <1>     ;
0000CC12 FE0D[13830100]     <1>     dec     byte [timer_events] ; 02/01/2017
                            <1>     ;
0000CC18 EBE6               <1>     jmp     short sysexit_8
                            <1>
                            <1> sysexit_11:
                            <1>     ;sub    ax, ax ; 0 ; 26/02/2017
                            <1>     ; 30/07/2022
0000CC1A 29C0               <1>     sub     eax, eax ; 0
                            <1> sysexit_12:
                            <1>     ; 26/02/2017 (Unlink IRQ callbacks belong to the user)
0000CC1C 803D[AE8E0100]00   <1>     cmp     byte [u.irqc], 0 ; Count of IRQ callbacks
0000CC23 7E53               <1>     jng     short sysexit_16 ; zero or invalid
                            <1>     ; 28/02/2017
                            <1>     ; clear IRQ callback flags (for 'sysrele' and 'sysret')
0000CC25 A2[AF8E0100]       <1>     mov     [u.irqwait], al ; 0 ; force to clear waiting flag
0000CC2A A2[B08E0100]       <1>     mov     [u.r_lock], al ; 0 ; force to clear busy flag
0000CC2F BE[60880100]       <1>     mov     esi, IRQ.owner
                            <1> sysexit_13:
0000CC34 AC                 <1>     lodsb
0000CC35 3A05[858E0100]     <1>     cmp     al, [u.uno] ; owner = current user ?
0000CC3B 7531               <1>     jne     short sysexit_14
0000CC3D C646FF00           <1>     mov     byte [esi-1], 0 ; owner = 0 : Free
                            <1>
                            <1>     ;;;
                            <1>     ; 22/08/2024 - Reset Hardware Interrupt Handler
0000CC41 89F1               <1>     mov     ecx, esi
0000CC43 81E9[60880100]     <1>     sub     ecx, IRQ.owner
                            <1>     ; CL = (user configurable) IRQ index + 1 (1 to 9)
0000CC49 B8[80350100]       <1>     mov     eax, IRQenum
                            <1> sysexit_18:
0000CC4E 3A08               <1>     cmp     cl, [eax]
0000CC50 740A               <1>     je      short sysexit_19
0000CC52 3D[8F350100]       <1>     cmp     eax, IRQenum+15
0000CC57 730D               <1>     jnb     short sysexit_20
0000CC59 40                 <1>     inc     eax
0000CC5A EBF2               <1>     jmp     short sysexit_18
                            <1> sysexit_19:
0000CC5C 2D[80350100]       <1>     sub     eax, IRQenum
                            <1>     ; AL = IRQ number (will be reset to system's IRQ handler)
                            <1>     ; AH = 0
                            <1>     ;mov    ah, 0 ; reset
0000CC61 E885510000         <1>     call    set_hardware_int_vector
                            <1> sysexit_20:
```

```
                            <1>     ;;;
                            <1>
0000CC66 FE0D[AE8E0100]     <1>     dec     byte [u.irqc]
0000CC6C 7408               <1>     jz      short sysexit_15
                            <1> sysexit_14:
0000CC6E 81FE[68880100]     <1>     cmp     esi, IRQ.owner + 8 ; the last IRQ index number ?
0000CC74 76BE               <1>     jna     short sysexit_13 ; no
                            <1> sysexit_15:
                            <1>     ;xor    al, al ; 0
                            <1>     ; 22/08/2024
0000CC76 31C0               <1>     xor     eax, eax
                            <1> sysexit_16: ; 2:
0000CC78 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
0000CC79 E8CE350000         <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
                            <1>     ;inc    al
                            <1>     ; 22/08/2024
0000CC7E 40                 <1>     inc     eax
                            <1>             ; inc r1 / increment file descriptor
                            <1>     ;cmp    ax, 10
0000CC7F 3C0A               <1>     cmp     al, 10
                            <1>             ; cmp r1,$10. / end of u.fp list?
0000CC81 72F6               <1>     jb      short sysexit_1
                            <1>             ; blt 1b / no, go back
                            <1>
                            <1>     ; 22/08/2024
                            <1>     ; reset IRQs if owned by the current (dying) process
0000CC83 BE[60880100]       <1>     mov     esi, IRQ.owner
                            <1>
                            <1>     ;movzx  ebx, byte [u.uno]
0000CC88 8A1D[858E0100]     <1>     mov     bl, [u.uno] ; 02/01/2017
                            <1>             ; movb  u.uno,r1 / yes, move dying process's number to r1
0000CC8E 88A3[678D0100]     <1>     mov     [ebx+p.stat-1], ah ; 0, SFREE
                            <1>             ; clrb p.stat-1(r1) / free the process
                            <1>     ; 10/04/2017
0000CC94 381D[DD880100]     <1>     cmp     [audio_user], bl
0000CC9A 7518               <1>     jne     short sysexit_17
                            <1>     ; reset audio device (current) owner and 'initializated' flag
0000CC9C 883D[DD880100]     <1>     mov     [audio_user], bh ; 0
                            <1>     ; 27/05/2017
0000CCA2 8B0D[C4880100]     <1>     mov     ecx, [audio_buffer]
0000CCA8 09C9               <1>     or      ecx, ecx
0000CCAA 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)
0000CCAC 29C9               <1>     sub     ecx, ecx
0000CCAE 890D[C4880100]     <1>     mov     [audio_buffer], ecx ; 0
                            <1>     ;pop    ebx
                            <1> sysexit_17:
                            <1>     ;shl    bx, 1
0000CCB4 D0E3               <1>     shl     bl, 1
                            <1>     ; 22/08/2024
                            <1>     ;shl    ebx, 1
                            <1>             ; asl r1 / use r1 for index into the below tables
0000CCB6 668B8B[068D0100]   <1>     mov     cx, [ebx+p.pid-2]
                            <1>             ; mov p.pid-2(r1),r3 / move dying process's name to r3
0000CCBD 668B93[268D0100]   <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
0000CCC4 30DB               <1>     xor     bl, bl ; 0
                            <1>             ; clr r2
0000CCC6 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
0000CCC8 80C302             <1>     add     bl, 2
                            <1>             ; add $2,r2 / search parent process table
                            <1>                         ; / for dying process's name
0000CCCB 66398B[268D0100]   <1>     cmp     [ebx+p.ppid-2], cx
                            <1>             ; cmp p.ppid-2(r2),r3 / found it?
0000CCD2 7513               <1>     jne     short sysexit_4
                            <1>             ; bne 3f / no
                            <1>     ;shr    bx, 1
0000CCD4 D0EB               <1>     shr     bl, 1
                            <1>             ; asr r2 / yes, it is a parent
0000CCD6 80BB[678D0100]03   <1>     cmp     byte [ebx+p.stat-1], 3 ; SZOMB
                            <1>             ; cmpb p.stat-1(r2),$3 / is the child of this
                            <1>                                    ; / dying process a zombie
0000CCDD 7506               <1>     jne     short sysexit_3
                            <1>             ; bne 2f / no
0000CCDF 88A3[678D0100]     <1>     mov     [ebx+p.stat-1], ah ; 0, SFREE
                            <1>             ; clrb p.stat-1(r2) / yes, free the child process
                            <1> sysexit_3: ; 2:
                            <1>     ;shr    bx, 1
0000CCE5 D0E3               <1>     shl     bl, 1
                            <1>             ; asl r2
                            <1> sysexit_4: ; 3:
                            <1>             ; / search the process name table
                            <1>             ; / for the dying process's parent
0000CCE7 663993[068D0100]   <1>     cmp     [ebx+p.pid-2], dx
                            <1>             ; cmp p.pid-2(r2),r4 / found it?
0000CCEE 7502               <1>     jne     short sysexit_5
                            <1>             ; bne 3f / no
0000CCF0 89DE               <1>     mov     esi, ebx
                            <1>             ; mov r2,r5 / yes, put index to p.pid table (parents
                            <1>                         ; / process # x2) in r5
                            <1> sysexit_5: ; 3:
                            <1>     ;cmp    bx, nproc + nproc
0000CCF2 80FB20             <1>     cmp     bl, nproc + nproc
                            <1>             ; cmp r2,$nproc+nproc / has whole table been searched?
0000CCF5 72D1               <1>     jb      short sysexit_2
                            <1>             ; blt 1b / no, go back
                            <1>             ; mov r5,r1 / yes, r1 now has parents process # x2
0000CCF7 21F6               <1>     and     esi, esi ; r5=r1
0000CCF9 743F               <1>     jz      short sysexit_6
                            <1>             ; beq 2f / no parent has been found.
                            <1>                    ; / The process just dies
                            <1>     ;shr    si, 1
                            <1>     ; 23/07/2022
```

```
0000CCFB D1EE                <1>     shr     esi, 1
                             <1>             ; asr r1 / set up index to p.stat
0000CCFD 8A86[678D0100]      <1>     mov     al, [esi+p.stat-1]
                             <1>             ; movb p.stat-1(r1),r2 / move status of parent to r2
0000CD03 20C0                <1>     and     al, al
0000CD05 7433                <1>     jz      short sysexit_6
                             <1>             ; beq 2f / if its been freed, 2f
0000CD07 3C03                <1>     cmp     al, 3
                             <1>             ; cmp r2,$3 / is parent a zombie?
0000CD09 742F                <1>     je      short sysexit_6
                             <1>             ; beq 2f / yes, 2f
                             <1>
                             <1>     ; BH = 0
0000CD0B 8A1D[858E0100]      <1>     mov     bl, [u.uno]
                             <1>             ; movb u.uno,r3 / move dying process's number to r3
0000CD11 C683[678D0100]03    <1>     mov     byte [ebx+p.stat-1], 3 ; SZOMB
                             <1>             ; movb $3,p.stat-1(r3) / make the process a zombie
                             <1>
0000CD18 3C01                <1>     cmp     al, 1 ; SRUN
0000CD1A 741E                <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
0000CD1C C686[678D0100]01    <1>     mov     byte [esi+p.stat-1], 1 ; SRUN
                             <1>     ;;;
                             <1>     ; 18/08/2024 - exit code
0000CD23 A0[B28E0100]        <1>     mov     al, [u.exit] ; exit code of the child
                             <1>     ; 20/08/2024
0000CD28 8883[078E0100]      <1>     mov     [ebx+p.exitc-1], al ; save it to use by the parent
                             <1>     ;;;
                             <1>     ;dec    byte [esi+p.stat-1]
                             <1>             ; decb  p.stat-1(r1) / awaken it by putting it (parent)
                             <1>     ;;;
                             <1>     ;mov    ax, si ; r1  (process number in AL)
                             <1>     ; 18/08/2024
0000CD2E 89F0                <1>     mov     eax, esi
                             <1>     ;;;
                             <1>     ;mov    ebx, runq + 4
                             <1>             ; mov $runq+4,r2 / on the runq
0000CD30 BB[248E0100]        <1>     mov     ebx, runq+2 ; normal run queue ; 02/01/2017
0000CD35 E8DC3C0000          <1>     call    putlu
                             <1>             ; jsr r0, putlu
                             <1> sysexit_6:
                             <1>             ; / the process dies
0000CD3A C605[858E0100]00    <1>     mov     byte [u.uno], 0
                             <1>             ; clrb u.uno / put zero as the process number,
                             <1>                ; / so "swap" will
0000CD41 E8D33B0000          <1>     call    swap
                             <1>             ; jsr r0,swap / overwrite process with another process
                             <1> hlt_sys:
                             <1>     ;sti
                             <1> hlts0:
0000CD46 F4                  <1>     hlt
0000CD47 EBFD                <1>     jmp     short hlts0
                             <1>             ; 0 / and thereby kill it; halt?
                             <1>
                             <1> syswait: ; < wait for a processs to die >
                             <1>     ; 20/08/2024
                             <1>     ; 30/07/2022
                             <1>     ; 23/07/2022 - TRDOS 386 v2.0.5
                             <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:
0000CD49 0FB61D[858E0100]    <1>     movzx   ebx, byte [u.uno] ; 01/09/2015
                             <1>             ; movb u.uno,r1 / put parents process number in r1
0000CD50 D0E3                <1>     shl     bl, 1
                             <1>     ;shl    bx, 1
                             <1>             ; asl r1 / x2 to get index into p.pid table
0000CD52 668B83[068D0100]    <1>     mov     ax, [ebx+p.pid-2]
                             <1>             ; mov p.pid-2(r1),r1 / get the name of this process
0000CD59 31F6                <1>     xor     esi, esi
                             <1>             ; clr r2
0000CD5B 31C9                <1>     xor     ecx, ecx ; 30/10/2013
                             <1>     ;xor    cl, cl
                             <1>             ; clr r3 / initialize reg 3
                             <1> syswait_1: ; 1:
                             <1>     ;add    si, 2
                             <1>     ; 23/07/2022
0000CD5D 46                  <1>     inc     esi
0000CD5E 46                  <1>     inc     esi
                             <1>             ; add $2,r2 / use r2 for index into p.ppid table
                             <1>                     ; / search table of parent processes
                             <1>                     ; / for this process name
0000CD5F 663B86[268D0100]    <1>     cmp     ax, [esi+p.ppid-2]
                             <1>             ; cmp p.ppid-2(r2),r1 / r2 will contain the childs
```

```
1085                            <1>                                     ; / process number
1086 0000CD66 7542              <1>     jne     short syswait_3
1087                            <1>             ;bne 3f / branch if no match of parent process name
1088                            <1>     ;inc    cx
1089 0000CD68 FEC1              <1>     inc     cl
1090                            <1>             ;inc r3 / yes, a match, r3 indicates number of children
1091                            <1>     ;shr    si, 1
1092                            <1>     ; 23/07/2022
1093 0000CD6A D1EE              <1>     shr     esi, 1
1094                            <1>             ; asr r2 / r2/2 to get index to p.stat table
1095                            <1>     ; The possible states ('p.stat' values) of a process are:
1096                            <1>     ;       0 = free or unused
1097                            <1>     ;       1 = active
1098                            <1>     ;       2 = waiting for a child process to die
1099                            <1>     ;       3 = terminated, but not yet waited for (zombie).
1100 0000CD6C 80BE[678D0100]03  <1>     cmp     byte [esi+p.stat-1], 3 ; SZOMB, 05/02/2014
1101                            <1>             ; cmpb p.stat-1(r2),$3 / is the child process a zombie?
1102 0000CD73 7533              <1>     jne     short syswait_2
1103                            <1>             ; bne 2f / no, skip it
1104 0000CD75 88BE[678D0100]    <1>     mov     [esi+p.stat-1], bh ; 0
1105                            <1>             ; clrb p.stat-1(r2) / yes, free it
1106                            <1>
1107                            <1>     ; 20/08/2024 (ebx = child's exit code)
1108 0000CD7B 31C0              <1>     xor     eax, eax
1109 0000CD7D 8A86[078E0100]    <1>     mov     al, [esi+p.exitc-1] ; exit code
1110 0000CD83 8B2D[308E0100]    <1>     mov     ebp, [u.usp]
1111                            <1>     ; 22/08/2024
1112 0000CD89 894510            <1>     mov     [ebp+16], eax ; ebx
1113                            <1>
1114                            <1>     ;shl    si, 1
1115                            <1>     ; 23/07/2022
1116 0000CD8C D1E6              <1>     shl     esi, 1
1117                            <1>             ; asl r2 / r2x2 to get index into p.pid table
1118                            <1>     ;movzx  eax, word [esi+p.pid-2]
1119                            <1>     ; 20/08/2024
1120 0000CD8E 668B86[068D0100]  <1>     mov     ax, [esi+p.pid-2]
1121 0000CD95 A3[348E0100]      <1>     mov     [u.r0], eax
1122                            <1>             ; mov p.pid-2(r2),*u.r0
1123                            <1>                             ; / put childs process name in (u.r0)
1124                            <1>     ;
1125                            <1>     ; Retro UNIX 386 v1 modification ! (17/09/2015)
1126                            <1>     ;
1127                            <1>     ; Parent process ID -p.ppid- field (of the child process)
1128                            <1>     ; must be cleared in order to prevent infinitive 'syswait'
1129                            <1>     ; system call loop from the application/program if it calls
1130                            <1>     ; 'syswait' again (mistakenly) while there is not a zombie
1131                            <1>     ; or running child process to wait. ('forktest.s', 17/09/2015)
1132                            <1>     ;
1133                            <1>     ; Note: syswait will return with error if there is not a
1134                            <1>     ;       zombie or running process to wait.
1135                            <1>
1136                            <1>     ;sub    ax, ax
1137                            <1>     ; 30/07/2022
1138 0000CD9A 29C0              <1>     sub     eax, eax ; 0
1139 0000CD9C 668986[268D0100]  <1>     mov     [esi+p.ppid-2], ax ; 0 ; 17/09/2015
1140 0000CDA3 E9D8FCFFFF        <1>     jmp     sysret0 ; ax = 0
1141                            <1>     ;
1142                            <1>     ;jmp    sysret
1143                            <1>             ; br sysret1 / return cause child is dead
1144                            <1> syswait_2: ; 2:
1145                            <1>     ;shl    si, 1
1146                            <1>     ; 23/07/2022
1147 0000CDA8 D1E6              <1>     shl     esi, 1
1148                            <1>             ; asl r2 / r2x2 to get index into p.ppid table
1149                            <1> syswait_3: ; 3:
1150 0000CDAA 6683FE20          <1>     cmp     si, nproc+nproc
1151                            <1>             ; cmp r2,$nproc+nproc / have all processes been checked?
1152 0000CDAE 72AD              <1>     jb      short syswait_1
1153                            <1>             ; blt 1b / no, continue search
1154                            <1>     ;and    cx, cx
1155 0000CDB0 20C9              <1>     and     cl, cl
1156                            <1>             ; tst r3 / one gets here if there are no children
1157                            <1>                      ; / or children that are still active
1158                            <1>     ; 30/10/2013
1159 0000CDB2 7515              <1>     jnz     short syswait_4
1160                            <1>     ;jz     error
1161                            <1>             ; beq error1 / there are no children, error
1162 0000CDB4 890D[348E0100]    <1>     mov     [u.r0], ecx ; 0
1163                            <1>
1164                            <1>     ; 20/08/2024 (ebx = child's exit code)
1165 0000CDBA 8B2D[308E0100]    <1>     mov     ebp, [u.usp]
1166 0000CDC0 49                <1>     dec     ecx ; -1 ; 0FFFFFFFFh
1167                            <1>     ; 22/08/2024
1168 0000CDC1 894D10            <1>     mov     [ebp+16], ecx ; ebx
1169                            <1>
1170 0000CDC4 E995FCFFFF        <1>     jmp     error
1171                            <1> syswait_4:
1172 0000CDC9 8A1D[858E0100]    <1>     mov     bl, [u.uno]
1173                            <1>             ; movb u.uno,r1 / there are children so put
1174                            <1>                             ; / parent process number in r1
1175 0000CDCF FE83[678D0100]    <1>     inc     byte [ebx+p.stat-1] ; 2, SWAIT, 05/02/2014
1176                            <1>             ; incb p.stat-1(r1) / it is waiting for
1177                            <1>                                 ; / other children to die
1178                            <1>     ; 04/11/2013
1179 0000CDD5 E83F3B0000        <1>     call    swap
1180                            <1>             ; jsr r0,swap / swap it out, because it's waiting
1181                            <1>
1182                            <1>     ; 23/08/2024 - bugfix
1183                            <1>     ; restore [u.usp]
1184                            <1>     ; (swap above changes [u.usp] just before wswap)
1185 0000CDDA 8925[308E0100]    <1>     mov     [u.usp], esp
1186                            <1>             ; points to user's regs on top ofn system stack
1187                            <1>
1188 0000CDE0 E964FFFFFF        <1>     jmp     syswait_0
1189                            <1>             ; br syswait / wait on next process
1190                            <1>
1191                            <1> sysfork: ; < create a new process >
1192                            <1>     ; 19/08/2024 - TRDOS 386 Kernel v2.0.9 (STDIN/STDOUT)
1193                            <1>     ; 23/07/2022 - TRDOS 386 Kernel v2.0.5
1194                            <1>     ; 02/01/2017 (TRDOS 386 modification)
1195                            <1>     ; 04/09/2015 - 18/05/2015
1196                            <1>     ; 28/08/2015 - 01/09/2015 - 02/09/2015
1197                            <1>     ; 09/05/2015 - 10/05/2015 - 14/05/2015
1198                            <1>     ; 06/05/2015 (Retro UNIX 386 v1 - Beginning)
1199                            <1>     ; 24/05/2013 - 14/02/2014 (Retro UNIX 8086 v1)
1200                            <1>     ;
1201                            <1>     ; 'sysfork' creates a new process. This process is referred
1202                            <1>     ; to as the child process. This new process core image is
1203                            <1>     ; a copy of that of the caller of 'sysfork'. The only
1204                            <1>     ; distinction is the return location and the fact that (u.r0)
1205                            <1>     ; in the old process (parent) contains the process id (p.pid)
1206                            <1>     ; of the new process (child). This id is used by 'syswait'.
1207                            <1>     ; 'sysfork' works in the following manner:
1208                            <1>     ;    1) The process status table (p.stat) is searched to find
```

```
1209                                    <1>     ;       a process number that is unused. If none are found
1210                                    <1>     ;       an error occurs.
1211                                    <1>     ;    2) when one is found, it becomes the child process number
1212                                    <1>     ;       and it's status (p.stat) is set to active.
1213                                    <1>     ;    3) If the parent had a control tty, the interrupt
1214                                    <1>     ;       character in that tty buffer is cleared.
1215                                    <1>     ;    4) The child process is put on the lowest priority run
1216                                    <1>     ;       queue via 'putlu'.
1217                                    <1>     ;    5) A new process name is gotten from 'mpid' (actually
1218                                    <1>     ;       it is a unique number) and is put in the child's unique
1219                                    <1>     ;       identifier; process id (p.pid).
1220                                    <1>     ;    6) The process name of the parent is then obtained and
1221                                    <1>     ;       placed in the unique identifier of the parent process
1222                                    <1>     ;       name is then put in 'u.r0'.
1223                                    <1>     ;    7) The child process is then written out on disk by
1224                                    <1>     ;       'wswap',i.e., the parent process is copied onto disk
1225                                    <1>     ;       and the child is born. (The child process is written
1226                                    <1>     ;       out on disk/drum with 'u.uno' being the child process
1227                                    <1>     ;       number.)
1228                                    <1>     ;    8) The parent process number is then restored to 'u.uno'.
1229                                    <1>     ;    9) The child process name is put in 'u.r0'.
1230                                    <1>     ;   10) The pc on the stack sp + 18 is incremented by 2 to
1231                                    <1>     ;       create the return address for the parent process.
1232                                    <1>     ;   11) The 'u.fp' list as then searched to see what files
1233                                    <1>     ;       the parent has opened. For each file the parent has
1234                                    <1>     ;       opened, the corresponding 'fsp' entry must be updated
1235                                    <1>     ;       to indicate that the child process also has opened
1236                                    <1>     ;       the file. A branch to 'sysret' is then made.
1237                                    <1>     ;
1238                                    <1>     ; Calling sequence:
1239                                    <1>     ;       from shell ?
1240                                    <1>     ; Arguments:
1241                                    <1>     ;       -
1242                                    <1>     ; Inputs: -
1243                                    <1>     ; Outputs: *u.r0 - child process name
1244                                    <1>     ; ..................................................................
1245                                    <1>     ;
1246                                    <1>     ; Retro UNIX 8086 v1 modification:
1247                                    <1>     ;       AX = r0 = PID (>0) (at the return of 'sysfork')
1248                                    <1>     ;       = process id of child a parent process returns
1249                                    <1>     ;       = process id of parent when a child process returns
1250                                    <1>     ;
1251                                    <1>     ;        In original UNIX v1, sysfork is called and returns as
1252                                    <1>     ;       in following manner: (with an example: c library, fork)
1253                                    <1>     ;
1254                                    <1>     ;       1:
1255                                    <1>     ;               sys     fork
1256                                    <1>     ;                       br 1f  / child process returns here
1257                                    <1>     ;               bes     2f     / parent process returns here
1258                                    <1>     ;               / pid of new process in r0
1259                                    <1>     ;               rts     pc
1260                                    <1>     ;       2: / parent process condionally branches here
1261                                    <1>     ;               mov     $-1,r0 / pid = -1 means error return
1262                                    <1>     ;               rts     pc
1263                                    <1>     ;
1264                                    <1>     ;       1: / child process brances here
1265                                    <1>     ;               clr     r0   / pid = 0 in child process
1266                                    <1>     ;               rts     pc
1267                                    <1>     ;
1268                                    <1>     ;       In UNIX v7x86 (386) by Robert Nordier (1999)
1269                                    <1>     ;               // pid = fork();
1270                                    <1>     ;               //
1271                                    <1>     ;               // pid == 0 in child process;
1272                                    <1>     ;               // pid == -1 means error return
1273                                    <1>     ;               // in child,
1274                                    <1>     ;               //      parents id is in par_uid if needed
1275                                    <1>     ;
1276                                    <1>     ;               _fork:
1277                                    <1>     ;                       mov     $.fork,eax
1278                                    <1>     ;                       int     $0x30
1279                                    <1>     ;                       jmp     1f
1280                                    <1>     ;                       jnc     2f
1281                                    <1>     ;                       jmp     cerror
1282                                    <1>     ;               1:
1283                                    <1>     ;                       mov     eax,_par_uid
1284                                    <1>     ;                       xor     eax,eax
1285                                    <1>     ;               2:
1286                                    <1>     ;                       ret
1287                                    <1>     ;
1288                                    <1>     ;       In Retro UNIX 8086 v1,
1289                                    <1>     ;       'sysfork' returns in following manner:
1290                                    <1>     ;
1291                                    <1>     ;               mov     ax, sys_fork
1292                                    <1>     ;               mov     bx, offset @f ; routine for child
1293                                    <1>     ;               int     20h
1294                                    <1>     ;               jc      error
1295                                    <1>     ;
1296                                    <1>     ;       ; Routine for parent process here (just after 'jc')
1297                                    <1>     ;                mov     word ptr [pid_of_child], ax
1298                                    <1>     ;                jmp     next_routine_for_parent
1299                                    <1>     ;
1300                                    <1>     ;       @@: ; routine for child process here
1301                                    <1>     ;            ....
1302                                    <1>     ;       NOTE: 'sysfork' returns to specified offset
1303                                    <1>     ;             for child process by using BX input.
1304                                    <1>     ;            (at first, parent process will return then
1305                                    <1>     ;            child process will return -after swapped in-
1306                                    <1>     ;            'syswait' is needed in parent process
1307                                    <1>     ;            if return from child process will be waited for.)
1308                                    <1>     ;
1309                                    <1>
1310                                    <1> ; / create a new process
1311                                    <1>     ; EBX = return address for child process
1312                                    <1>         ; (Retro UNIX 8086 v1 modification !)
1313 0000CDE5 31F6                      <1>     xor     esi, esi
1314                                    <1>             ; clr r1
1315                                    <1> sysfork_1: ; 1: / search p.stat table for unused process number
1316 0000CDE7 46                       <1>     inc     esi
1317                                    <1>             ; inc r1
1318 0000CDE8 80BE[678D0100]00         <1>     cmp     byte [esi+p.stat-1], 0 ; SFREE, 05/02/2014
1319                                    <1>             ; tstb p.stat-1(r1) / is process active, unused, dead
1320 0000CDEF 760B                     <1>     jna     short sysfork_2
1321                                    <1>             ; beq 1f / it's unused so branch
1322 0000CDF1 6683FE10                 <1>     cmp     si, nproc
1323                                    <1>             ; cmp r1,$nproc / all processes checked
1324 0000CDF5 72F0                     <1>     jb      short sysfork_1
1325                                    <1>             ; blt 1b / no, branch back
1326                                    <1>     ;
1327                                    <1>     ; Retro UNIX 8086 v1. modification:
1328                                    <1>     ;       Parent process returns from 'sysfork' to address
1329                                    <1>     ;       which is just after 'sysfork' system call in parent
1330                                    <1>     ;       process. Child process returns to address which is put
1331                                    <1>     ;       in BX register by parent process for 'sysfork'.
1332                                    <1>     ;
```

```
1333                                    <1>             ; add $2,18.(sp) / add 2 to pc when trap occured, points
1334                                    <1>                             ; / to old process return
1335                                    <1>             ; br error1 / no room for a new process
1336                                    <1> sysfork_err:
1337 0000CDF7 E962FCFFFF                <1>     jmp     error
1338                                    <1> sysfork_2: ; 1:
1339 0000CDFC E8258AFFFF                <1>     call    allocate_page
1340                                    <1>     ;jc     error
1341                                    <1>     ; 23/07/2022
1342 0000CE01 72F4                      <1>     jc      short sysfork_err
1343 0000CE03 50                        <1>     push    eax   ; UPAGE (user structure page) address
1344                                    <1>     ; Retro UNIX 386 v1 modification!
1345 0000CE04 E80D8CFFFF                <1>     call    duplicate_page_dir
1346                                    <1>             ; EAX = New page directory
1347 0000CE09 730B                      <1>     jnc     short sysfork_3
1348 0000CE0B 58                        <1>     pop     eax   ; UPAGE (user structure page) address
1349 0000CE0C E8D48BFFFF                <1>     call    deallocate_page
1350 0000CE11 E948FCFFFF                <1>     jmp     error
1351                                    <1> sysfork_3:
1352                                    <1>     ; Retro UNIX 386 v1 modification !
1353 0000CE16 56                        <1>     push    esi
1354 0000CE17 E88A3B0000                <1>     call    wswap ; save current user (u) structure, user registers
1355                                    <1>                   ; and interrupt return components (for IRET)
1356 0000CE1C 8705[8C8E0100]            <1>     xchg    eax, [u.pgdir] ; page directory of the child process
1357 0000CE22 A3[908E0100]              <1>     mov     [u.ppgdir], eax ; page directory of the parent process
1358 0000CE27 5E                        <1>     pop     esi
1359 0000CE28 58                        <1>     pop     eax   ; UPAGE (user structure page) address
1360                                    <1>             ; [u.usp] = esp
1361 0000CE29 89F7                      <1>     mov     edi, esi
1362                                    <1>     ;shl    di, 2
1363                                    <1>     ; 23/07/2022
1364 0000CE2B C1E702                    <1>     shl     edi, 2
1365 0000CE2E 8987[748D0100]            <1>     mov     [edi+p.upage-4], eax ; memory page for 'user' struct
1366 0000CE34 A3[888E0100]              <1>     mov     [u.upage], eax ; memory page for 'user' struct (child)
1367                                    <1>     ;;;
1368                                    <1>     ; 19/08/2024 - reset STDIN/STDOUT redirections (and ungetchar)
1369 0000CE39 31C0                      <1>     xor     eax, eax ; 0
1370                                    <1>     ; 19/08/2024
1371                                    <1>     ;mov    [u.stdin], al ; 0
1372                                    <1>     ;mov    [u.stdout], al ; 0
1373                                    <1>     ;mov    [u.ungetc], al ; 0
1374                                    <1>     ;mov    [u.getc], al ; 0
1375 0000CE3B A3[9C8E0100]              <1>     mov     [u.stdin], eax ; 0
1376                                    <1>     ;;;
1377                                    <1>
1378                                    <1>     ; 28/08/2015
1379                                    <1>     ;movzx  eax, byte [u.uno] ; parent process number
1380                                    <1>     ; 19/08/2024
1381                                    <1>     ; eax = 0
1382 0000CE40 A0[858E0100]              <1>     mov     al, [u.uno] ; parent process number
1383                                    <1>             ; movb u.uno,-(sp) / save parent process number
1384 0000CE45 89C7                      <1>     mov     edi, eax
1385 0000CE47 50                        <1>         push      eax ; **
1386 0000CE48 8A87[478D0100]            <1>     mov     al, [edi+p.ttyc-1] ; console tty (parent)
1387                                    <1>     ; 18/09/2015
1388                                    <1>     ;;mov   [esi+p.ttyc-1], al ; set child's console tty
1389                                    <1>     ;;mov   [esi+p.waitc-1], ah ; 0 ; reset child's wait channel
1390                                    <1>     ;mov    [esi+p.ttyc-1], ax ; al - set child's console tty
1391                                    <1>                                ; ah - reset child's wait channel
1392                                    <1>     ; 23/07/2022
1393 0000CE4E 8886[478D0100]            <1>     mov     [esi+p.ttyc-1], al ; set child's console tty
1394 0000CE54 89F0                      <1>     mov     eax, esi
1395 0000CE56 A2[858E0100]              <1>     mov     [u.uno], al ; child process number
1396                                    <1>             ;movb r1,u.uno / set child process number to r1
1397 0000CE5B FE86[678D0100]            <1>         inc     byte [esi+p.stat-1] ; 1, SRUN, 05/02/2014
1398                                    <1>             ; incb p.stat-1(r1) / set p.stat entry for child
1399                                    <1>                                 ; / process to active status
1400                                    <1>             ; mov u.ttyp,r2 / put pointer to parent process'
1401                                    <1>                               ; / control tty buffer in r2
1402                                    <1>                 ; beq 2f / branch, if no such tty assigned
1403                                    <1>             ; clrb 6(r2) / clear interrupt character in tty buffer
1404                                    <1>     ; 2:
1405 0000CE61 53                        <1>     push    ebx  ; * return address for the child process
1406                                    <1>                  ; * Retro UNIX 8086 v1 feature only !
1407                                    <1>     ; (Retro UNIX 8086 v1 modification!)
1408                                    <1>             ; mov $runq+4,r2
1409 0000CE62 BB[248E0100]              <1>     mov     ebx, runq+2 ; normal run queue ; 02/01/2017
1410 0000CE67 E8AA3B0000                <1>     call    putlu
1411                                    <1>             ; jsr r0,putlu / put child process on lowest priority
1412                                    <1>                            ; / run queue
1413                                    <1>     ; 23/07/2022
1414 0000CE6C D1E6                      <1>     shl     esi, 1
1415                                    <1>     ;shl    si, 1
1416                                    <1>             ; asl r1 / multiply r1 by 2 to get index
1417                                    <1>                      ; / into p.pid table
1418 0000CE6E 66FF05[1C8E0100]          <1>     inc     word [mpid]
1419                                    <1>             ; inc mpid / increment m.pid; get a new process name
1420 0000CE75 66A1[1C8E0100]            <1>     mov     ax, [mpid]
1421 0000CE7B 668986[068D0100]          <1>     mov     [esi+p.pid-2], ax
1422                                    <1>             ;mov mpid,p.pid-2(r1) / put new process name
1423                                    <1>                                   ; / in child process' name slot
1424 0000CE82 5A                        <1>     pop     edx  ; * return address for the child process
1425                                    <1>                  ; * Retro UNIX 8086 v1 feature only !
1426 0000CE83 5B                        <1>     pop     ebx  ; **
1427                                    <1>     ;mov    ebx, [esp] ; ** parent process number
1428                                    <1>             ; movb (sp),r2 / put parent process number in r2
1429                                    <1>     ; 23/07/2022
1430 0000CE84 D1E3                      <1>     shl     ebx, 1
1431                                    <1>     ;shl    bx, 1
1432                                    <1>             ;asl r2 / multiply by 2 to get index into below tables
1433                                    <1>     ;movzx eax, word [ebx+p.pid-2]
1434 0000CE86 668B83[068D0100]          <1>     mov     ax, [ebx+p.pid-2]
1435                                    <1>             ; mov p.pid-2(r2),r2 / get process name of parent
1436                                    <1>                                  ; / process
1437 0000CE8D 668986[268D0100]          <1>     mov     [esi+p.ppid-2], ax
1438                                    <1>             ; mov r2,p.ppid-2(r1) / put parent process name
1439                                    <1>                                   ; / in parent process slot for child
1440 0000CE94 A3[348E0100]              <1>     mov     [u.r0], eax
1441                                    <1>             ; mov r2,*u.r0 / put parent process name on stack
1442                                    <1>                            ; / at location where r0 was saved
1443 0000CE99 8B2D[2C8E0100]            <1>     mov     ebp, [u.sp] ; points to return address (EIP for IRET)
1444 0000CE9F 895500                    <1>     mov     [ebp], edx ; *, CS:EIP -> EIP
1445                                    <1>                        ; * return address for the child process
1446                                    <1>             ; mov $sysret1,-(sp) /
1447                                    <1>             ; mov sp,u.usp / contents of sp at the time when
1448                                    <1>                            ; / user is swapped out
1449                                    <1>             ; mov $sstack,sp / point sp to swapping stack space
1450                                    <1>     ; 04/09/2015 - 01/09/2015
1451                                    <1>     ; [u.usp] = esp
1452 0000CEA2 68[7ECA0000]              <1>     push    sysret ; ***
1453 0000CEA7 8925[308E0100]            <1>     mov     [u.usp], esp ; points to 'sysret' address (***)
1454                                    <1>                          ; (for child process)
1455 0000CEAD 31C0                      <1>     xor     eax, eax
1456 0000CEAF 66A3[688E0100]            <1>     mov     [u.ttyp], ax ; 0
```

```
1457                                <1>     ;
1458 0000CEB5 E8EC3A0000          <1>     call    wswap ; Retro UNIX 8086 v1 modification !
1459                                <1>             ; jsr r0,wswap / put child process out on drum
1460                                <1>             ; jsr r0,unpack / unpack user stack
1461                                <1>             ; mov u.usp,sp / restore user stack pointer
1462                                <1>             ; tst (sp)+ / bump stack pointer
1463                                <1>     ; Retro UNIX 386 v1 modification !
1464 0000CEBA 58                    <1>     pop     eax ; ***
1465                                <1>     ;shl    bx, 1
1466                                <1>     ; 23/07/2022
1467 0000CEBB D1E3                <1>     shl     ebx, 1
1468 0000CEBD 8B83[748D0100]      <1>     mov     eax, [ebx+p.upage-4] ; UPAGE address ; 14/05/2015
1469 0000CEC3 E8163B0000          <1>     call    rswap ; restore parent process 'u' structure,
1470                                <1>                     ; registers and return address (for IRET)
1471                                <1>             ;movb (sp)+,u.uno / put parent process number in u.uno
1472 0000CEC8 0FB705[1C8E0100]    <1>         movzx   eax, word [mpid]
1473 0000CECF A3[348E0100]        <1>     mov     [u.r0], eax
1474                                <1>             ; mov mpid,*u.r0 / put child process name on stack
1475                                <1>                              ; / where r0 was saved
1476                                <1>             ; add $2,18.(sp) / add 2 to pc on stack; gives parent
1477                                <1>                              ; / process return
1478                                <1>     ;xor    ebx, ebx
1479 0000CED4 31F6                <1>     xor     esi, esi
1480                                <1>             ;clr r1
1481                                <1> sysfork_4: ; 1: / search u.fp list to find the files
1482                                <1>           ; / opened by the parent process
1483                                <1>     ; 01/09/2015
1484                                <1>     ;xor    bh, bh
1485                                <1>     ;mov    bl, [esi+u.fp]
1486 0000CED6 8A86[3E8E0100]        <1>     mov     al, [esi+u.fp]
1487                                <1>             ; movb u.fp(r1),r2 / get an open file for this process
1488                                <1>         ;or bl, bl
1489 0000CEDC 08C0                <1>     or      al, al
1490 0000CEDE 7406                <1>     jz      short sysfork_5
1491                                <1>             ; beq 2f / file has not been opened by parent,
1492                                <1>                    ; / so branch
1493                                <1>     ;mov    ah, 10 ; Retro UNIX 386 v1 fsp structure size = 10 bytes
1494                                <1>     ;mul    ah
1495                                <1>     ; 23/07/2022
1496                                <1>     ; Retro UNIX 386 v2 & TRDOS 386 v2.0.5 fsp struc size = 16 bytes
1497                                <1>     ;mov    ebx, eax
1498                                <1>     ;shl    ebx, 4 ; * 16
1499                                <1>     ;inc    byte [ebx+fsp-10]
1500                                <1>
1501                                <1>     ; 23/07/2022 (BugFix)
1502 0000CEE0 FE80[24840100]        <1>     inc     byte [OF_OPENCOUNT+eax]
1503                                <1>
1504                                <1>     ;;movzx ebx, ax
1505                                <1>     ;mov    bx, ax
1506                                <1>     ;shl    bx, 3
1507                                <1>             ; asl r2 / multiply by 8
1508                                <1>                    ; asl r2 / to get index into fsp table
1509                                <1>                    ; asl r2
1510                                <1>     ;inc    byte [ebx+fsp-2]
1511                                <1>             ; incb fsp-2(r2) / increment number of processes
1512                                <1>                            ; / using file, because child will now be
1513                                <1>                            ; / using this file
1514                                <1> sysfork_5: ; 2:
1515 0000CEE6 46                    <1>         inc     esi
1516                                <1>             ; inc r1 / get next open file
1517                                <1>     ; 23/07/2022
1518 0000CEE7 6683FE20            <1>     cmp     si, OPENFILES ; = 10
1519                                <1>     ;cmp    si, 10
1520                                <1>             ; cmp r1,$10. / 10. files is the maximum number which
1521                                <1>                          ; / can be opened
1522 0000CEEB 72E9                <1>     jb      short sysfork_4
1523                                <1>             ; blt 1b / check next entry
1524 0000CEED E98CFBFFFF          <1>     jmp     sysret
1525                                <1>             ; br sysret1
1526                                <1>
1527                                <1> syscreat: ; < create file >
1528                                <1>     ; 29/08/2024
1529                                <1>     ; 25/08/2024 - TRDOS 386 v2.0.9
1530                                <1>     ; 08/08/2022
1531                                <1>     ; 23/07/2022 - TRDOS 386 Kernel v2.0.5
1532                                <1>     ; 13/11/2017
1533                                <1>     ; 27/10/2016
1534                                <1>     ; 25/10/2016 - 26/10/2016
1535                                <1>     ; 15/10/2016 - 16/10/2016 - 17/10/2016
1536                                <1>     ; 10/10/2016 (TRDOS 386 = TRDOS v2.0)
1537                                <1>     ;            -derived from INT_21H.ASM-
1538                                <1>     ;             ("loc_INT21h_create_file")
1539                                <1>         ;   10/07/2011 (12/03/2011)
1540                                <1>         ;   INT 21h Function AH = 3Ch
1541                                <1>         ;   Create File
1542                                <1>         ;   INPUT
1543                                <1>         ;      CX = Attributes
1544                                <1>         ;          DS:DX= Address of zero terminaned path name
1545                                <1>         ;
1546                                <1>     ; 27/12/2015 (Retro UNIX 386 v1.1)
1547                                <1>     ; 14/05/2015 (Retro UNIX 386 v1 - Beginning)
1548                                <1>     ; 27/05/2013 (Retro UNIX 8086 v1)
1549                                <1>     ;
1550                                <1>     ; 'syscreat' called with two arguments; name and mode.
1551                                <1>     ; u.namep points to name of the file and mode is put
1552                                <1>     ; on the stack. 'namei' is called to get i-number of the file.
1553                                <1>     ; If the file aready exists, it's mode and owner remain
1554                                <1>     ; unchanged, but it is truncated to zero length. If the file
1555                                <1>     ; did not exist, an i-node is created with the new mode via
1556                                <1>     ; 'maknod' whether or not the file already existed, it is
1557                                <1>     ; open for writing. The fsp table is then searched for a free
1558                                <1>     ; entry. When a free entry is found, proper data is placed
1559                                <1>     ; in it and the number of this entry is put in the u.fp list.
1560                                <1>     ; The index to the u.fp (also know as the file descriptor)
1561                                <1>     ; is put in the user's r0.
1562                                <1>     ;
1563                                <1>     ; Calling sequence:
1564                                <1>     ;       syscreate; name; mode
1565                                <1>     ; Arguments:
1566                                <1>     ;       name - name of the file to be created
1567                                <1>     ;       mode - mode of the file to be created
1568                                <1>     ; Inputs: (arguments)
1569                                <1>     ; Outputs: *u.r0 - index to u.fp list
1570                                <1>     ;                  (the file descriptor of new file)
1571                                <1>     ; ..........................................................
1572                                <1>     ;
1573                                <1>     ; Retro UNIX 8086 v1 modification:
1574                                <1>     ;       'syscreate' system call has two arguments; so,
1575                                <1>     ;       * 1st argument, name is pointed to by BX register
1576                                <1>     ;       * 2nd argument, mode is in CX register
1577                                <1>     ;
1578                                <1>     ;       AX register (will be restored via 'u.r0') will return
1579                                <1>     ;       to the user with the file descriptor/number
1580                                <1>     ;       (index to u.fp list).
```

```
1581                            <1>     ;
1582                            <1>     ;call  arg2
1583                            <1>     ; * name - 'u.namep' points to address of file/path name
1584                            <1>     ;          in the user's program segment ('u.segmnt')
1585                            <1>     ;          with offset in BX register (as sysopen argument 1).
1586                            <1>     ; * mode - sysopen argument 2 is in CX register
1587                            <1>     ;          which is on top of stack.
1588                            <1>     ;
1589                            <1>     ; TRDOS 386 (10/10/2016)
1590                            <1>     ;
1591                            <1>        ; INPUT ->
1592                            <1>        ;     CL = File Attributes
1593                            <1>     ;             bit 0 (1) - Read only file (R)
1594                            <1>     ;              bit 1 (1) - Hidden file (H)
1595                            <1>        ;              bit 2 (1) - System file (R)
1596                            <1>     ;              bit 3 (1) - Volume label/name (V)
1597                            <1>        ;              bit 4 (1) - Subdirectory (D)
1598                            <1>     ;             bit 5 (1) - File has been archived (A)
1599                            <1>        ;           EBX = Pointer to filename (ASCIIZ) -path-
1600                            <1>     ;
1601                            <1>     ; OUTPUT ->
1602                            <1>     ;          eax = File/Device Handle/Number (index) (AL)
1603                            <1>     ;          cf = 1 -> Error code in AL
1604                            <1>     ;
1605                            <1>     ; Modified Registers: EAX (at the return of system call)
1606                            <1>     ;
1607                            <1>     ; Note: If the file is existing and it has not any one
1608                            <1>     ;       of S,H,R,V,D attributes, it will be truncated
1609                            <1>     ;       to zero length; otherwise, access error will be
1610                            <1>     ;       returned.
1611                            <1>
1612                            <1> sysmkdir_0:
1613 0000CEF2 F6C108             <1>     test   cl, 08h ; Volume name
1614 0000CEF5 740A             <1>     jz     short syscreat_0
1615                            <1>
1616                            <1>     ; Volume name or long name creation
1617                            <1>     ; is not permitted (in TRDOS 386)!
1618 0000CEF7 B80B000000       <1>     mov    eax, ERR_FILE_ACCESS  ; 11 ; 'permission denied !'
1619 0000CEFC E9D8000000       <1>         jmp sysopen_dev_err ; 08/08/2022
1620                            <1> syscreat_0:
1621                            <1>         ;mov       [u.namep], ebx
1622 0000CF01 51               <1>     push   ecx
1623 0000CF02 89DE             <1>     mov    esi, ebx
1624                            <1>     ; file name is forced, change directory as temporary
1625                            <1>     ;mov   ax, 1
1626                            <1>     ;mov   [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
1627                            <1>     ;call  set_working_path
1628 0000CF04 E89B4D0000       <1>     call   set_working_path_x ; 17/10/2016
1629                            <1>     ;jc    short syscreat_err
1630                            <1>     ; 23/07/2022
1631 0000CF09 7305             <1>     jnc    short syscreat_3
1632 0000CF0B E9E2000000       <1>     jmp    syscreat_err
1633                            <1>
1634                            <1> syscreat_3:
1635                            <1>     ; 16/10/2016
1636 0000CF10 803D[3B830100]00 <1>     cmp    byte [SWP_inv_fname], 0
1637 0000CF17 7776             <1>     ja     short syscreat_inv_fname ; invalid file name !
1638                            <1>
1639                            <1>     ; Here, we have a valid path and also a valid file name
1640                            <1>     ; (Working dir has been changed if the path
1641                            <1>     ;  -file name string- had contained a dir name.)
1642                            <1>
1643                            <1>     ;xor   ax, ax
1644                            <1>     ; 25/08/2024
1645 0000CF19 31C0             <1>     xor    eax, eax
1646                            <1>     ;mov   esi, FindFile_Name
1647 0000CF1B E84ABAFFFF       <1>     call   find_first_file
1648 0000CF20 59               <1>     pop    ecx
1649                            <1>            ; ESI = Directory Entry (FindFile_DirEntry) Location
1650                            <1>            ; EDI = Directory Buffer Directory Entry Location
1651                            <1>            ; EAX = File Size
1652                            <1>            ;  BL = Attributes of The File/Directory
1653                            <1>            ;  BH = Long Name Yes/No Status (>0 is YES)
1654                            <1>            ;  DX > 0 : Ambiguous filename chars are used
1655 0000CF21 7274             <1>     jc     short syscreat_1 ; file not found (the good!)
1656                            <1>                             ; or another error (the bad!)
1657                            <1>
1658                            <1>     ; (& the uggly!) truncate file to zero length before open
1659                            <1>
1660                            <1>     ;'*' and '?' already checked at 'set_working_path' stage
1661                            <1>     ;and   dx, dx
1662                            <1>     ;jnz   short sysmkdir_err ; permission denied
1663                            <1>                               ; invalid filename chars
1664                            <1>
1665                            <1>     ;test  cl, 10h ; subdirectory ?
1666                            <1>     ;jnz   short sysmkdir_err
1667                            <1>
1668                            <1>     ; BL = File Attributes:
1669                            <1>     ;             bit 0 (1) - Read only file (R)
1670                            <1>     ;              bit 1 (1) - Hidden file (H)
1671                            <1>        ;              bit 2 (1) - System file (R)
1672                            <1>     ;              bit 3 (1) - Volume label/name (V)
1673                            <1>        ;              bit 4 (1) - Subdirectory (D)
1674                            <1>     ;             bit 5 (1) - File has been archived
1675                            <1>
1676                            <1>     ; * existing directory must not be truncated
1677                            <1>     ;   (we don't know it is empty or not, at this stage)
1678                            <1>     ; * existing volume name (or a long name) can not be
1679                            <1>     ;   re-created or truncated by 'syscreat'
1680                            <1>     ; * A file with S, H, R attributes must not be truncated
1681                            <1>     ;   (change attributes to normal, if you need truncate it)
1682                            <1>
1683 0000CF23 F6C31F             <1>     test   bl, 00011111b  ; check attributes of existing file
1684 0000CF26 7559             <1>     jnz    short sysmkdir_err
1685                            <1>
1686                            <1>     ;; normal file, OK to continue...
1687                            <1>
1688                            <1>     ; ESI = FindFile_DirEntry
1689 0000CF28 668B4614         <1>     mov    ax, [esi+DirEntry_FstClusHI] ; 20
1690 0000CF2C C1E010           <1>     shl    eax, 16 ; 13/11/2017
1691 0000CF2F 668B461A         <1>     mov    ax, [esi+DirEntry_FstClusLO] ; 26
1692                            <1>
1693                            <1>     ; 24/08/2024
1694 0000CF33 21C0             <1>     and    eax, eax
1695 0000CF35 7426             <1>     jz     short skip_truncate ; first cluster is 0 !
1696                            <1>
1697                            <1>     ; EAX = First cluster to be truncated/unlinked
1698 0000CF37 57               <1>     push   edi
1699 0000CF38 51               <1>     push   ecx
1700 0000CF39 BE00010900       <1>     mov    esi, Logical_DOSDisks
1701 0000CF3E 29C9             <1>     sub    ecx, ecx
1702                            <1>     ;mov   ch, [Current_Drv] ; = [FindFile_Drv]
1703                            <1>     ; 03/09/2024
1704 0000CF40 8A2D[E27F0100]   <1>     mov    ch, [FindFile_Drv]
```

```
1705 0000CF46 01CE                <1>     add     esi, ecx
1706                              <1>     ; ESI = Logical dos drive description table address
1707 0000CF48 E8C1F7FFFF          <1>     call    truncate_cluster_chain
1708 0000CF4D 59                  <1>     pop     ecx
1709 0000CF4E 5F                  <1>     pop     edi
1710 0000CF4F 7237                <1>     jc      short syscreate_truncate_err
1711                              <1>
1712                              <1>     ; 03/09/2024
1713 0000CF51 66C747140000        <1>     mov     word [edi+DirEntry_FstClusHI], 0
1714 0000CF57 66C7471A0000        <1>     mov     word [edi+DirEntry_FstClusLO], 0
1715                              <1>
1716                              <1>     ; 24/08/2024
1717                              <1> skip_truncate:
1718                              <1>     ; 26/10/2016
1719                              <1>     ; EDI = Directory entry address in directory buffer
1720                              <1>     ; Update directory entry
1721 0000CF5D E88ADDFFFF          <1>     call    convert_current_date_time
1722                              <1>     ; OUTPUT -> DX = Date in dos dir entry format
1723                              <1>         ;       AX = Time in dos dir entry format
1724 0000CF62 66894716            <1>     mov     [edi+DirEntry_WrtTime], ax
1725 0000CF66 66895718            <1>     mov     [edi+DirEntry_WrtDate], dx
1726 0000CF6A 66895712            <1>     mov     [edi+DirEntry_LastAccDate], dx
1727 0000CF6E 31C0                <1>     xor     eax, eax ; file size = 0
1728 0000CF70 89471C              <1>     mov     [edi+DirEntry_FileSize], eax ; 0
1729 0000CF73 C605[697E0100]02    <1>     mov     byte [DirBuff_ValidData], 2 ; data changed sign
1730                              <1>     ;mov    esi, FindFile_DirEntry
1731                              <1>     ; 03/09/2024
1732 0000CF7A B201                <1>     mov     dl, 1 ; open file for writing
1733 0000CF7C E9AB000000          <1>     jmp     sysopen_2 ; 08/08/2022
1734                              <1>
1735                              <1> sysmkdir_err:
1736                              <1>     ; 1 = write, 2 = read & write, >2 = invalid
1737 0000CF81 B80B000000          <1>         mov eax, ERR_FILE_ACCESS  ; 11 ; 'permission denied !'
1738 0000CF86 EB74                <1>         jmp short sysopen_err
1739                              <1>
1740                              <1> syscreate_truncate_err:
1741 0000CF88 B812000000          <1>     mov     eax, ERR_DRV_WRITE ; 18 ; 'disk write error !'
1742 0000CF8D EB6D                <1>         jmp short sysopen_err
1743                              <1>
1744                              <1> syscreat_inv_fname:  ; invalid file name chars
1745                              <1>     ; 16/10/2016
1746 0000CF8F B81A000000          <1>     mov     eax, ERR_INV_FILE_NAME  ; 26 ; invalid file name chars
1747 0000CF94 59                  <1>     pop     ecx
1748 0000CF95 EB65                <1>     jmp     short sysopen_err
1749                              <1>
1750                              <1> syscreat_1:
1751                              <1>     ; Error code in EAX
1752 0000CF97 3C02                <1>         cmp al, 02h ; 'File not found' error
1753 0000CF99 7561                <1>         jne short sysopen_err
1754                              <1>
1755 0000CF9B F6C110              <1>     test    cl, 10h ; Directory
1756                              <1>     ;jnz    sysmkdir_2
1757                              <1>     ; 23/07/2022
1758 0000CF9E 7405                <1>     jz      short syscreat_2
1759 0000CFA0 E9C3010000          <1>     jmp     sysmkdir_2
1760                              <1>
1761                              <1> syscreat_2:
1762 0000CFA5 BE[24800100]        <1>     mov     esi, FindFile_Name
1763                              <1>         ;xor       edx, edx
1764 0000CFAA 31C0                <1>         xor eax, eax ; File Size  = 0
1765 0000CFAC 31DB                <1>     xor     ebx, ebx
1766 0000CFAE 4B                  <1>     dec     ebx ; FFFFFFFFh -> create empty file
1767                              <1>                 ;              (only for FAT fs)
1768                              <1>     ; CL = File Attributes
1769 0000CFAF E8AFECFFFF          <1>     call    create_file
1770 0000CFB4 7246                <1>     jc      short sysopen_err
1771                              <1>             ; EAX = New file's first cluster
1772                              <1>             ; ESI = Logical Dos Drv Descr. Table Addr.
1773                              <1>             ; EBX = offset CreateFile_Size
1774                              <1>             ; ECX = Sectors per cluster (<256)
1775                              <1>             ; EDX = Directory entry index/number (<65536)
1776                              <1>             ; 29/08/2024
1777                              <1>             ; EBX = File Size (0 for a new, empty file)
1778                              <1>             ; ECX = Directory Entry Index/Number (<2048)
1779                              <1>             ;       (in directory cluster, not in directory)
1780                              <1>             ; EDX = Directory Cluster Number (of the file)
1781                              <1>
1782                              <1>     ; 26/10/2016
1783                              <1>     ;mov    esi, Directory_Buffer
1784                              <1>     ;shl    dx, 5 ; *32
1785                              <1>     ;add    esi, edx
1786                              <1>     ;; esi = directory entry address in directory buffer
1787                              <1>
1788                              <1>     ; Here, directory entry has been created but last
1789                              <1>     ; modification date & time of the parent dir has not
1790                              <1>     ; been updated, yet!
1791                              <1>     ; (Note: Directory and FAT buffers have been updated...)
1792                              <1>
1793 0000CFB6 E84FDEFFFF          <1>     call    update_parent_dir_lmdt ; now, it is OK too!
1794                              <1>
1795                              <1>     ; 25/10/2016
1796 0000CFBB 66B80018            <1>     mov     ax, 1800h
1797 0000CFBF BE[24800100]        <1>     mov     esi, FindFile_Name
1798 0000CFC4 E8A1B9FFFF          <1>     call    find_first_file
1799 0000CFC9 7231                <1>     jc      short sysopen_err
1800                              <1>
1801                              <1>     ; Only possible error after here is
1802                              <1>     ; "too many open files !" error.
1803                              <1>     ;
1804                              <1>     ; If "syscreat" will return with that error,
1805                              <1>     ; (the file has been created but it could not be opened)
1806                              <1>     ; the user must retry to open this file again
1807                              <1>     ; or must close another file before using
1808                              <1>     ; "sysopen" system call.
1809                              <1>
1810 0000CFCB B201                <1>     mov     dl, 1 ; open file for writing
1811                              <1>     ; ESI = Directory Entry (FindFile_DirEntry) Location
1812                              <1>     ; EAX = File Size (= 0)
1813 0000CFCD EB5D                <1>     jmp     short sysopen_2
1814                              <1>
1815                              <1> sysopen: ;<open file>
1816                              <1>     ; 19/12/2025 - TRDOS 386 v2.0.10
1817                              <1>     ; 03/09/2024
1818                              <1>     ; 19/08/2024 - TRDOS 386 v2.0.9
1819                              <1>     ; 23/07/2022 - TRDOS 386 v2.0.5
1820                              <1>     ; 17/04/2021 - TRDOS 386 v2.0.4
1821                              <1>     ;      (temporary modifications)
1822                              <1>     ; 26/10/2016
1823                              <1>     ; 24/10/2016
1824                              <1>     ; 17/10/2016
1825                              <1>     ; 15/10/2016
1826                              <1>     ; 06/10/2016, 07/10/2016, 08/10/2016
1827                              <1>     ; 05/10/2016 (TRDOS 386 = TRDOS v2.0)
1828                              <1>     ;           -derived from INT_21H.ASM-
```

```
1829                            <1>     ;           ("loc_INT21h_open_file")
1830                            <1>         ;   26/02/2011
1831                            <1>         ;   INT 21h Function AH = 3Dh
1832                            <1>         ;   Open File
1833                            <1>         ;   INPUT
1834                            <1>         ;      AL= File Access Value
1835                            <1>     ;            0- Open for reading
1836                            <1>     ;             1- Open for writing
1837                            <1>         ;             2- Open for reading and writing
1838                            <1>         ;          DS:DX= Pointer to filename (ASCIIZ)
1839                            <1>         ;
1840                            <1>     ; 14/05/2015 (Retro UNIX 386 v1 - Beginning)
1841                            <1>     ; 22/05/2013 - 27/05/2013 (Retro UNIX 8086 v1)
1842                            <1>     ;
1843                            <1>     ; 'sysopen' opens a file in following manner:
1844                            <1>     ;    1) The second argument in a sysopen says whether to
1845                            <1>     ;       open the file ro read (0) or write (>0).
1846                            <1>     ;    2) I-node of the particular file is obtained via 'namei'.
1847                            <1>     ;    3) The file is opened by 'iopen'.
1848                            <1>     ;    4) Next housekeeping is performed on the fsp table
1849                            <1>     ;       and the user's open file list - u.fp.
1850                            <1>     ;       a) u.fp and fsp are scanned for the next available slot.
1851                            <1>     ;       b) An entry for the file is created in the fsp table.
1852                            <1>     ;       c) The number of this entry is put on u.fp list.
1853                            <1>     ;       d) The file descriptor index to u.fp list is pointed
1854                            <1>     ;          to by u.r0.
1855                            <1>     ;
1856                            <1>     ; Calling sequence:
1857                            <1>     ;       sysopen; name; mode
1858                            <1>     ; Arguments:
1859                            <1>     ;       name - file name or path name
1860                            <1>     ;       mode - 0 to open for reading
1861                            <1>     ;              1 to open for writing
1862                            <1>     ; Inputs: (arguments)
1863                            <1>     ; Outputs: *u.r0 - index to u.fp list (the file descriptor)
1864                            <1>     ;                  is put into r0's location on the stack.
1865                            <1>     ; ..........................................................
1866                            <1>     ;
1867                            <1>     ; Retro UNIX 8086 v1 modification:
1868                            <1>     ;       'sysopen' system call has two arguments; so,
1869                            <1>     ;       * 1st argument, name is pointed to by BX register
1870                            <1>     ;       * 2nd argument, mode is in CX register
1871                            <1>     ;
1872                            <1>     ;       AX register (will be restored via 'u.r0') will return
1873                            <1>     ;       to the user with the file descriptor/number
1874                            <1>     ;       (index to u.fp list).
1875                            <1>     ;
1876                            <1>     ;call  arg2
1877                            <1>     ; * name - 'u.namep' points to address of file/path name
1878                            <1>     ;          in the user's program segment ('u.segmnt')
1879                            <1>     ;          with offset in BX register (as sysopen argument 1).
1880                            <1>     ; * mode - sysopen argument 2 is in CX register
1881                            <1>     ;          which is on top of stack.
1882                            <1>     ;
1883                            <1>     ; jsr r0,arg2 / get sys args into u.namep and on stack
1884                            <1>     ;
1885                            <1>             ; system call registers: ebx, ecx (through 'sysenter')
1886                            <1>     ;
1887                            <1>     ; TRDOS 386 (05/10/2016)
1888                            <1>     ;
1889                            <1>         ; INPUT ->
1890                            <1>         ;      CL = File Access Value (Open Mode)
1891                            <1>     ;             0 - Open file for reading
1892                            <1>     ;              1 - Open file for writing
1893                            <1>         ;             2 - Open device for reading
1894                            <1>     ;             3 - Open device for writing
1895                            <1>         ;           EBX = Pointer to filename/devicename (ASCIIZ)
1896                            <1>     ; OUTPUT ->
1897                            <1>     ;          eax = File/Device Handle/Number (index) (AL)
1898                            <1>     ;          cf = 1 -> Error code in AL
1899                            <1>     ;
1900                            <1>     ; Modified Registers: EAX (at the return of system call)
1901                            <1>     ;
1902                            <1>
1903 0000CFCF 80F901           <1>     cmp     cl, 1 ; read file (0), write file (1)
1904 0000CFD2 7614             <1>     jna     short sysopen_0
1905                            <1>
1906                            <1>     ; 17/04/2021 (temporary)
1907                            <1>     ;cmp    cl, 3
1908                            <1>     ;jna    sysopen_device
1909                            <1>
1910                            <1>     ; Invalid access code
1911 0000CFD4 B817000000       <1>     mov     eax, ERR_INV_PARAMETER
1912                            <1>     ;jmp    sysopen_dev_err
1913                            <1>
1914                            <1> sysopen_dev_err:
1915 0000CFD9 A3[348E0100]     <1>     mov     [u.r0], eax
1916 0000CFDE A3[A08E0100]     <1>     mov     [u.error], eax
1917 0000CFE3 E976FAFFFF       <1>     jmp     error
1918                            <1>
1919                            <1> sysopen_0:
1920                            <1>     ;mov    [u.namep], ebx
1921 0000CFE8 51               <1>     push    ecx
1922 0000CFE9 89DE             <1>     mov     esi, ebx
1923                            <1>     ; file name is forced, change directory as temporary
1924                            <1>     ;mov    ax, 1
1925                            <1>     ;mov    [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
1926                            <1>     ;call   set_working_path
1927 0000CFEB E8B44C0000       <1>     call    set_working_path_x ; 17/10/2016
1928 0000CFF0 731E             <1>     jnc     short sysopen_1
1929                            <1>
1930                            <1> syscreat_err: ; ecx = file attributes (for 'syscreat')
1931 0000CFF2 59               <1>     pop     ecx ; open mode
1932 0000CFF3 21C0             <1>     and     eax, eax  ; 0 -> Bad Path!
1933 0000CFF5 7505             <1>     jnz     short sysopen_err
1934                            <1>     ; eax = 0
1935 0000CFF7 B80C000000       <1>     mov     eax, ERR_DIR_NOT_FOUND ; Directory not found !
1936                            <1> sysopen_err:
1937 0000CFFC A3[348E0100]     <1>     mov     [u.r0], eax
1938 0000D001 A3[A08E0100]     <1>     mov     [u.error], eax
1939 0000D006 E86E4D0000       <1>     call    reset_working_path
1940 0000D00B E94EFAFFFF       <1>     jmp     error
1941                            <1>
1942                            <1> sysopen_1:
1943                            <1>     ;mov    esi, FindFile_Name
1944 0000D010 66B80018         <1>     mov     ax, 1800h ; Only files
1945 0000D014 E851B9FFFF       <1>     call    find_first_file
1946 0000D019 5A               <1>     pop     edx
1947 0000D01A 72E0             <1>     jc      short sysopen_err ; eax = 2 (File not found !)
1948                            <1>
1949                            <1>     ; check_open_file_attr_access_code
1950                            <1>
1951 0000D01C F6C307           <1>     test    bl, 7  ; system, hidden, readonly
1952 0000D01F 740B             <1>     jz      short sysopen_2
```

```
                                <1>
0000D021 20D2                   <1>     and     dl, dl ; 0 = read mode
0000D023 7407                   <1>     jz      short sysopen_2
                                <1>
                                <1> sysopen_access_err:
                                <1>     ; 1 = write, 2 = read & write, >2 = invalid
0000D025 B80B000000             <1>     mov     eax, ERR_FILE_ACCESS ; 11 = 'permission denied !'
0000D02A EBD0                   <1>     jmp     short sysopen_err
                                <1>
                                <1> sysopen_2:
                                <1>     ; esi = Directory Entry (FindFile_DirEntry) Location
                                <1>     ;mov    ebx, esi
                                <1>     ; 03/09/2024
0000D02C 89FB                   <1>     mov     ebx, edi ; Directory entry in directory buffer
0000D02E 31F6                   <1>     xor     esi, esi ; 0
0000D030 31FF                   <1>     xor     edi, edi ; 0
                                <1> sysopen_3: ; scan the list of entries in fsp table
0000D032 80BE[3E8E0100]00       <1>     cmp     byte [esi+u.fp], 0
0000D039 760E                   <1>     jna     short sysopen_4 ; empty slot
                                <1>     ;inc    si
                                <1>     ; 19/08/2024
0000D03B 46                     <1>     inc     esi
0000D03C 6683FE0A               <1>     cmp     si, 10
0000D040 72F0                   <1>     jb      short sysopen_3
                                <1> toomanyf:
0000D042 B80D000000             <1>     mov     eax, ERR_TOO_MANY_FILES ; too many open files !
0000D047 EBB3                   <1>     jmp     short sysopen_err
                                <1>
                                <1> sysopen_4:
0000D049 80BF[E4830100]00       <1>     cmp     byte [edi+OF_MODE], 0 ; Scan open files table
0000D050 7609                   <1>     jna     short sysopen_5
                                <1>
                                <1>     ;inc    di
                                <1>     ; 19/08/2024
0000D052 47                     <1>     inc     edi
0000D053 6683FF20               <1>     cmp     di, OPENFILES ; max. number of open files (sytem)
0000D057 72F0                   <1>     jb      short sysopen_4
0000D059 EBE7                   <1>     jmp     short toomanyf
                                <1>
                                <1> sysopen_5:
                                <1>     ; 19/08/2024
                                <1>     ;;;
0000D05B 50                     <1>     push    eax ; size
0000D05C 668B4314               <1>     mov     ax, [ebx+DirEntry_FstClusHI]
0000D060 C1E010                 <1>     shl     eax, 16
0000D063 668B431A               <1>     mov     ax, [ebx+DirEntry_FstClusLO]
0000D067 5B                     <1>     pop     ebx ; size
                                <1>     ; eax = First Cluster
                                <1>     ; ebx = File Size
0000D068 80FA01                 <1>     cmp     dl, 1 ; is open mode = open for write ?
0000D06B 7535                   <1>     jne     short sysopen_8
                                <1>     ; check if the file is already open for write
0000D06D 57                     <1>     push    edi
0000D06E 29FF                   <1>     sub     edi, edi
0000D070 8A15[E27F0100]         <1>     mov     dl, [FindFile_Drv]
                                <1> sysopen_5_@:
0000D076 80BF[E4830100]02       <1>     cmp     byte [edi+OF_MODE], 2 ; open for write
0000D07D 7519                   <1>     jne     short sysopen_5_@@@
0000D07F 3A97[C4830100]         <1>     cmp     dl, [edi+OF_DRIVE]
0000D085 7511                   <1>     jne     short sysopen_5_@@@
0000D087 C1E702                 <1>     shl     edi, 2
0000D08A 3B87[44830100]         <1>     cmp     eax, [edi+OF_FCLUSTER]
0000D090 7503                   <1>     jne     short sysopen_5_@@
0000D092 5F                     <1>     pop     edi
0000D093 EB90                   <1>     jmp     short sysopen_access_err
                                <1> sysopen_5_@@:
0000D095 C1EF02                 <1>     shr     edi, 2
                                <1>     ;;;
                                <1> sysopen_5_@@@:
0000D098 47                     <1>     inc     edi
0000D099 6683FF20               <1>     cmp     di, OPENFILES
0000D09D 72D7                   <1>     jb      short sysopen_5_@
0000D09F 5F                     <1>     pop     edi
0000D0A0 B201                   <1>     mov     dl, 1 ; open for write
                                <1> sysopen_8:
0000D0A2 FEC2                   <1>     inc     dl
0000D0A4 8897[E4830100]         <1>     mov     [edi+OF_MODE], dl
0000D0AA 8A15[E27F0100]         <1>     mov     dl, [FindFile_Drv]
0000D0B0 8897[C4830100]         <1>     mov     [edi+OF_DRIVE], dl ; Logical DOS drive number
                                <1>     ;shl    di, 2 ; *4 (dword offset)
                                <1>     ; 23/07/2022
0000D0B6 C1E702                 <1>     shl     edi, 2
                                <1>
                                <1>     ; 19/08/2024
                                <1>     ;mov    [edi+OF_SIZE], eax ; File size in bytes
0000D0B9 899F[C4840100]         <1>     mov     [edi+OF_SIZE], ebx ; File size in bytes
                                <1>
                                <1>     ; 19/08/2024
                                <1>     ; eax = Fist Cluster
                                <1>     ;mov    ax, [ebx+DirEntry_FstClusHI]
                                <1>     ;shl    eax, 16
                                <1>     ;mov    ax, [ebx+DirEntry_FstClusLO]
                                <1>
0000D0BF 8987[44830100]         <1>     mov     [edi+OF_FCLUSTER], eax ; First cluster
0000D0C5 8987[C4860100]         <1>     mov     [edi+OF_CCLUSTER], eax ; Current cluster
                                <1>
0000D0CB 31DB                   <1>     xor     ebx, ebx
0000D0CD 899F[44840100]         <1>     mov     [edi+OF_POINTER], ebx ; offset pointer (0)
0000D0D3 899F[44870100]         <1>     mov     [edi+OF_CCINDEX], ebx ; cluster index (0)
                                <1>
0000D0D9 A1[54800100]           <1>     mov     eax, [FindFile_DirFirstCluster]
0000D0DE 8987[44850100]         <1>     mov     [edi+OF_DIRFCLUSTER], eax
                                <1>
0000D0E4 A1[58800100]           <1>     mov     eax, [FindFile_DirCluster]
0000D0E9 8987[C4850100]         <1>     mov     [edi+OF_DIRCLUSTER], eax
                                <1>
                                <1>     ; Get (& Save) Volume ID
                                <1>     ; Important for files of removable drives
                                <1>     ; (In order to check the drive has same volume/disk)
0000D0EF 88D7                   <1>     mov     bh, dl
0000D0F1 81C300010900           <1>     add     ebx, Logical_DOSDisks
                                <1>
0000D0F7 8A4303                 <1>     mov     al, [ebx+LD_FATType]
                                <1>
                                <1> ; 19/12/2025
                                <1> %if 0
                                <1>     cmp     al, 1
                                <1>     jb      short sysopen_6_fs
                                <1> %endif
0000D0FA 3C02                   <1>     cmp     al, 2
0000D0FC 7705                   <1>     ja      short sysopen_6_fat32
                                <1> sysopen_6_fat:
0000D0FE 8B432D                 <1>     mov     eax, [ebx+LD_BPB+VolumeID]
0000D101 EB03                   <1>     jmp     short sysopen_7
```

```
                                <1>
                                <1> ; 19/12/2025
                                <1> %if 0
                                <1> sysopen_6_fs:
                                <1>     mov     eax, [ebx+LD_FS_VolumeSerial]
                                <1>     jmp     short sysopen_7
                                <1> %endif
                                <1>
                                <1> sysopen_6_fat32:
0000D103 8B4349                 <1>     mov     eax, [ebx+LD_BPB+FAT32_VolID]
                                <1> sysopen_7:
0000D106 A3[38770100]           <1>     mov     [Current_VolSerial], eax
                                <1>
0000D10B 8987[44860100]         <1>     mov     [edi+OF_VOLUMEID], eax
                                <1>
                                <1>     ; 24/10/2016
                                <1>     ;shr   di, 1 ; 4/2, word offset
                                <1>     ; 23/07/2022
0000D111 D1EF                   <1>     shr     edi, 1
0000D113 668B1D[5C800100]       <1>     mov     bx, [FindFile_DirEntryNumber]
0000D11A 66899F[C4870100]       <1>     mov     [edi+OF_DIRENTRY], bx
                                <1>
0000D121 31D2                   <1>     xor     edx, edx
                                <1>     ;;shr   di, 2 ; /4 (byte offset)
                                <1>     ;shr   di, 1 ; 2/2, byte offset
                                <1>     ; 23/07/2022
0000D123 D1EF                   <1>     shr     edi, 1
0000D125 8897[24840100]         <1>     mov     byte [edi+OF_OPENCOUNT], dl ; 0
0000D12B 8897[04840100]         <1>     mov     byte [edi+OF_STATUS], dl ; 0
                                <1>
0000D131 89FB                   <1>     mov     ebx, edi
0000D133 FEC3                   <1>     inc     bl
                                <1>
0000D135 889E[3E8E0100]         <1>     mov     [esi+u.fp], bl ; Open File Entry Number
0000D13B 8935[348E0100]         <1>     mov     [u.r0], esi ; move index to u.fp list
                                <1>                         ; into eax on stack
                                <1>
0000D141 E8334C0000             <1>     call    reset_working_path
                                <1>
0000D146 E933F9FFFF             <1>     jmp     sysret
                                <1>
                                <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> ; 27/03/2020 - Retro UNIX 386 v2 - FSP (OPEN FILES) TABLE
                                <1>
                                <1> ;Entry
                                <1> ;           15                    7                    0
                                <1> ;  1       |---------------------------------------------|
                                <1> ;          |              i-number of open file            |
                                <1> ;          |---------------------------------------------|
                                <1> ;          |        high word of 32 bit i-number         |
                                <1> ;          |---------------------------------------------|
                                <1> ;          | open mode & status  |   device number       |
                                <1> ;          |---------------------------------------------|
                                <1> ;          |    reserved byte    |     open count        |
                                <1> ;          |---------------------------------------------|
                                <1> ;          | offset pointer, i.e., r/w pointer to file   |
                                <1> ;          |---------------------------------------------|
                                <1> ;          |   64 bit file offset pointer (bit 16-31)    |
                                <1> ;          |---------------------------------------------|
                                <1> ;          |   64 bit file offset pointer (bit 32-47)    |
                                <1> ;          |---------------------------------------------|
                                <1> ;          |   64 bit file offset pointer (bit 48-63)    |
                                <1> ;          |---------------------------------------------|
                                <1> ;  2       |                                             |
                                <1> ;          |---------------------------------------------|
                                <1> ;          |                                             |
                                <1> ;          |---------------------------------------------|
                                <1> ;          |                                             |
                                <1> ;          |---------------------------------------------|
                                <1> ;          |                                             |
                                <1> ;          |---------------------------------------------|
                                <1> ;          |                                             |
                                <1>
                                <1> ; 23/07/2022 - TRDOS 386 Kernel v2.0.5
                                <1> ; OPENFILES equ 10 (sysdefs.s)
                                <1> ;; 06/10/2016
                                <1> ;; Open File Parameters (trdoskx.s)
                                <1> ;OF_FCLUSTER:      resd OPENFILES  ; First clusters of open files
                                <1> ;OF_DRIVE:  resb OPENFILES  ; Logical DOS drive numbers of open files
                                <1> ;OF_MODE:   resb OPENFILES  ; Open mode (1 = read, 2 = write, 3 = r&w)
                                <1> ;OF_STATUS: resb OPENFILES  ; (bit 0 = read, bit 1 = write)
                                <1> ;OF_OPENCOUNT:     resb OPENFILES  ; Open counts of open files
                                <1> ;OF_POINTER:       resd OPENFILES ; File seek/read/write pointer
                                <1> ;OF_SIZE:   resd OPENFILES ; File sizes of open files (in bytes)
                                <1> ;OF_DIRFCLUSTER: resd OPENFILES  ; Directory First Clusters of open files
                                <1> ;OF_DIRCLUSTER:    resd OPENFILES  ; Directory (Entry) Clusters of open files
                                <1> ;OF_VOLUMEID:      resd OPENFILES  ; Vol ID for removable drives of open files
                                <1> ;OF_CCLUSTER:      resd OPENFILES  ; Current clusters of open files
                                <1> ;OF_CCINDEX:       resd OPENFILES  ; Cluster index numbers of current clusters
                                <1> ;; 24/10/2016
                                <1> ;OF_DIRENTRY:      resw OPENFILES  ; Directory entry index no. in dir cluster
                                <1>
                                <1> ; 17/04/2021
                                <1> ; ('sysopen_device' procedure is disabled as temporary)
                                <1>
                                <1> ;sysopen_device:
                                <1> ;   ; 15/10/2016
```

```
2201                    <1> ;   ; 08/10/2016
2202                    <1> ;   ; 07/10/2016 (TRDOS 386 = TRDOS v2.0)
2203                    <1> ;   push    ecx ; open mode
2204                    <1> ;   mov     ebp, esp
2205                    <1> ;   mov     ecx, 16 ; transfer length = 16 bytes
2206                    <1> ;   sub     esp, ecx
2207                    <1> ;   mov     edi, esp ; destination address
2208                    <1> ;   mov     esi, ebx ; dev name in user's memory space
2209                    <1> ;   call    transfer_from_user_buffer
2210                    <1> ;   jnc     short sysopen_dev_0
2211                    <1> ;   ; eax = ERR_OUT_OF_MEMORY = 4 = ERR_MINOR_IM
2212                    <1> ;   pop     ecx
2213                    <1> ;sysopen_dev_err:
2214                    <1> ;   mov     [u.r0], eax
2215                    <1> ;   mov     [u.error], eax
2216                    <1> ;   jmp     error
2217                    <1> ;sysopen_dev_0:
2218                    <1> ;   mov     esi, edi ; Device name addr (max. 16 bytes, ASCIIZ)
2219                    <1> ;                    ; for example: "tty, TTY, /dev/tty"
2220                    <1> ;   call    get_device_number
2221                    <1> ;   mov     esp, ebp
2222                    <1> ;   pop     ecx
2223                    <1> ;   jnc     short sysopen_dev_1
2224                    <1> ;   mov     eax, ERR_INV_DEV_NAME ; 24 ; 'invalid device name !'
2225                    <1> ;   jmp     short sysopen_dev_err
2226                    <1> ;sysopen_dev_1:
2227                    <1> ;   ; eax = Device Number (AL)
2228                    <1> ;   ;  cl = Open mode (2 = device read, 3 = device write)
2229                    <1> ;       xor     ebx, ebx ; 0
2230                    <1> ;sysopen_dev_2: ; scan the list of entries
2231                    <1> ;       cmp     [ebx+u.fp], bl ; 0
2232                    <1> ;       jna     short sysopen_dev_3 ; empty slot
2233                    <1> ;       inc     bl
2234                    <1> ;       cmp     bl, 10
2235                    <1> ;   jb      short sysopen_dev_2
2236                    <1> ;   ;
2237                    <1> ;   mov     eax, ERR_TOO_MANY_FILES ; too many open files !
2238                    <1> ;   jmp     short sysopen_dev_err
2239                    <1> ;sysopen_dev_3:
2240                    <1> ;   mov     [u.r0], ebx ; File/Device index/handle/descriptor
2241                    <1> ;   ; eax = device number (entry offset)
2242                    <1> ;   mov     ch, [eax+DEV_ACCESS] ; bit 0 = accessable by users
2243                    <1> ;                                ; bit 1 = read access perm
2244                    <1> ;                                ; bit 2 = write access perm
2245                    <1> ;                                ; bit 3 = IOCTL permit to users
2246                    <1> ;                                ; bit 4 = block device if set
2247                    <1> ;                                ; bit 5 = 16 bit or 1024 byte
2248                    <1> ;                                ; bit 6 = 32 bit or 2048 byte
2249                    <1> ;                                ; bit 7 = installable device drv
2250                    <1> ;   test    ch, 1 ; accessable by normal users (except root)
2251                    <1> ;   jnz     short sysopen_dev_4 ; yes, permission has been given
2252                    <1> ;   cmp     byte [u.uid], 0 ; root?
2253                    <1> ;   jna     short sysopen_dev_4 ; superuser can open all devices
2254                    <1> ;sysopen_dev_perm_err:
2255                    <1> ;   mov     eax, ERR_DEV_ACCESS  ; 11 = 'permission denied !'
2256                    <1> ;   jmp     short sysopen_dev_err
2257                    <1> ;sysopen_dev_4:
2258                    <1> ;   shr     ch, 1 ; result: 1 = read, 2 = write, 3 = r & w
2259                    <1> ;   dec     cl  ; result: 1 = read, 2 = write
2260                    <1> ;   test    cl, ch
2261                    <1> ;   jz      short sysopen_dev_perm_err
2262                    <1> ;
2263                    <1> ;   shl     ch, 1 ; bit 0 = 0
2264                    <1> ;   ; eax = device number (entry offset)
2265                    <1> ;   call    device_open
2266                    <1> ;   jc      short sysopen_dev_perm_err
2267                    <1> ;
2268                    <1> ;   ; eax = device number (entry offset)
2269                    <1> ;   or      al, 80h ; set device bit (set bit 7 to 1)
2270                    <1> ;   mov     ebx, [u.r0]
2271                    <1> ;   mov     [ebx+u.fp], al ; bit 7 (=1) points to device
2272                    <1> ;
2273                    <1> ;   jmp     sysret
2274                    <1>
2275                    <1> sysmkdir: ; < make directory >
2276                    <1>     ; 23/07/2022 - TRDOS 386 Kernel v2.0.5
2277                    <1>     ; 15/10/2016
2278                    <1>     ; 10/10/2016 (TRDOS 386 = TRDOS v2.0)
2279                    <1>     ;            -derived from INT_21H.ASM-
2280                    <1>     ;             ("loc_INT21h_create_file")
2281                    <1>         ;   10/07/2011 (12/03/2011)
2282                    <1>         ;   INT 21h Function AH = 3Ch
2283                    <1>         ;   Create File
2284                    <1>         ;   INPUT
2285                    <1>         ;      CX = Attributes
2286                    <1>         ;           DS:DX= Address of zero terminaned path name
2287                    <1>         ;
2288                    <1>           ;
2289                    <1>     ; 14/05/2015 (Retro UNIX 386 v1 - Beginning)
2290                    <1>     ; 27/05/2013 - 02/08/2013 (Retro UNIX 8086 v1)
2291                    <1>     ;
2292                    <1>     ; 'sysmkdir' creates an empty directory whose name is
2293                    <1>     ; pointed to by arg 1. The mode of the directory is arg 2.
2294                    <1>     ; The special entries '.' and '..' are not present.
2295                    <1>     ; Errors are indicated if the directory already exists or
2296                    <1>     ; user is not the super user.
2297                    <1>     ;
2298                    <1>     ; Calling sequence:
2299                    <1>     ;       sysmkdir; name; mode
2300                    <1>     ; Arguments:
2301                    <1>     ;       name - points to the name of the directory
2302                    <1>     ;       mode - mode of the directory
2303                    <1>     ; Inputs: (arguments)
2304                    <1>     ; Outputs: -
2305                    <1>     ;    (sets 'directory' flag to 1;
2306                    <1>     ;    'set user id on execution' and 'executable' flags to 0)
2307                    <1>     ; ..........................................................
2308                    <1>     ;
2309                    <1>     ; Retro UNIX 8086 v1 modification:
2310                    <1>     ;       'sysmkdir' system call has two arguments; so,
2311                    <1>     ;       * 1st argument, name is pointed to by BX register
2312                    <1>     ;       * 2nd argument, mode is in CX register
2313                    <1>     ;
2314                    <1>     ; TRDOS 386 (10/10/2016)
2315                    <1>     ;
2316                    <1>         ; INPUT ->
2317                    <1>         ;      CL = Directory Attributes
2318                    <1>     ;             bit 0 (1) - Read only file/dir (R)
2319                    <1>     ;              bit 1 (1) - Hidden file/dir (H)
2320                    <1>         ;             bit 2 (1) - System file/dir (R)
2321                    <1>     ;              bit 3 (1) - Volume label/name (V)
2322                    <1>         ;             bit 4 (1) - Subdirectory (D)
2323                    <1>     ;             bit 5 (1) - File/Dir has been archived (A)
2324                    <1>     ;          CX = 0 -> create normal directory
```

```
2325                            <1>         ;          EBX = Pointer to directory name (ASCIIZ) -path-
2326                            <1>     ;
2327                            <1>     ; OUTPUT ->
2328                            <1>     ;          eax = First cluster of the new directory
2329                            <1>     ;          cf = 1 -> Error code in AL
2330                            <1>     ;
2331                            <1>     ; Modified Registers: EAX (at the return of system call)
2332                            <1>     ;
2333                            <1>     ; Note: If the file or directory is existing
2334                            <1>     ;       an access error will be returned.
2335                            <1>
2336 0000D14B 6621C9            <1>     and     cx, cx ; if cx = 0 -> create a normal subdir
2337 0000D14E 7414              <1>     jz      short sysmkdir_1
2338                            <1>
2339 0000D150 F6C110            <1>     test    cl, 10h ; if dir flags set, also use other flags
2340                            <1>     ;jnz    sysmkdir_0 ; jump to head of 'syscreat'
2341                            <1>     ; 23/07/2022
2342 0000D153 7405              <1>     jz      short sysmkdir_invf
2343                            <1> sysmkdir_3:
2344 0000D155 E998FDFFFF        <1>     jmp     sysmkdir_0
2345                            <1>
2346                            <1> sysmkdir_invf:
2347                            <1>     ; CX has wrong flags
2348 0000D15A B817000000        <1>     mov     eax, ERR_INV_FLAGS
2349 0000D15F E975FEFFFF        <1>     jmp     sysopen_dev_err
2350                            <1>
2351                            <1> sysmkdir_1:
2352 0000D164 B110              <1>     mov     cl, 10h ; set subdir flag and reset other flags
2353                            <1>     ;jmp    sysmkdir_0
2354                            <1>     ; 23/07/2022
2355 0000D166 EBED              <1>     jmp     short sysmkdir_3 ; jump to head of 'syscreat'
2356                            <1> sysmkdir_2:
2357                            <1>     ; jump from 'syscreat' ; from 'syscreat_1'
2358                            <1>     ;  CL = Directory attributes/flags
2359 0000D168 BE[24800100]        <1>     mov     esi, FindFile_Name
2360 0000D16D E8DFD8FFFF        <1>     call    make_sub_directory
2361                            <1>     ;jc     sysopen_err       ; NOTE: Old type (TRDOS 8086)
2362                            <1>                               ; error codes must be modified
2363                            <1>                               ; for next TRDOS 386 versions
2364                            <1>                               ; (10/10/2016)
2365                            <1>                               ; Old (MSDOS type)
2366                            <1>                               ; error codes (2011):
2367                            <1>                               ;  2 = file not found
2368                            <1>                               ;  3 = directory not found
2369                            <1>                               ;  5 = access denied
2370                            <1>                               ; 12 = no more files
2371                            <1>                               ; 19 = disk write protected
2372                            <1>                               ; 39 = insufficient disk space
2373                            <1>                               ; 'sysdefs.s' ; 10/10/2016
2374                            <1>     ; 23/07/2022
2375 0000D172 7305              <1>     jnc     short sysmkdir_4
2376 0000D174 E983FEFFFF        <1>     jmp     sysopen_err
2377                            <1>
2378                            <1> sysmkdir_4:
2379 0000D179 A3[348E0100]      <1>     mov     [u.r0], eax ; New sub dir's first cluster
2380 0000D17E E8F64B0000        <1>         call        reset_working_path
2381 0000D183 E9F6F8FFFF        <1>     jmp     sysret
2382                            <1>
2383                            <1> sysclose: ;<close file>
2384                            <1>     ; 23/07/2022 - TRDOS 386 v2.0.5
2385                            <1>     ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
2386                            <1>     ;
2387                            <1>     ; 14/05/2015 (Retro UNIX 386 v1 - Beginning)
2388                            <1>     ; 22/05/2013 - 26/05/2013 (Retro UNIX 8086 v1)
2389                            <1>     ;
2390                            <1>     ; 'sysclose', given a file descriptor in 'u.r0', closes the
2391                            <1>     ; associated file. The file descriptor (index to 'u.fp' list)
2392                            <1>     ; is put in r1 and 'fclose' is called.
2393                            <1>     ;
2394                            <1>     ; Calling sequence:
2395                            <1>     ;       sysclose
2396                            <1>     ; Arguments:
2397                            <1>     ;       -
2398                            <1>     ; Inputs: *u.r0 - file descriptor
2399                            <1>     ; Outputs: -
2400                            <1>     ; ..........................................................
2401                            <1>     ;
2402                            <1>     ; Retro UNIX 8086 v1 modification:
2403                            <1>     ;       The user/application program puts file descriptor
2404                            <1>     ;        in BX register as 'sysclose' system call argument.
2405                            <1>     ;       (argument transfer method 1)
2406                            <1>
2407                            <1>     ; TRDOS 386 (06/10/2016)
2408                            <1>     ;
2409                            <1>         ; INPUT ->
2410                            <1>         ;      EBX = File Handle/Number (file index) (AL)
2411                            <1>     ; OUTPUT ->
2412                            <1>     ;          cf = 0 -> EAX = 0
2413                            <1>     ;          cf = 1 -> Error code in EAX (ERR_FILE_NOT_OPEN)
2414                            <1>     ;
2415                            <1>     ; Modified Registers: EAX (at the return of system call)
2416                            <1>     ;
2417                            <1>
2418 0000D188 89D8              <1>     mov     eax, ebx
2419 0000D18A 31DB              <1>     xor     ebx, ebx
2420 0000D18C 891D[348E0100]    <1>     mov     [u.r0], ebx ; 0  ; return value of EAX
2421 0000D192 E8B5300000        <1>     call    fclose
2422                            <1>     ;jnc    sysret
2423                            <1>     ; 23/07/2022
2424 0000D197 7205              <1>     jc      short sysclose_err
2425 0000D199 E9E0F8FFFF        <1>     jmp     sysret
2426                            <1> sysclose_err:
2427                            <1>     ; 24/04/2025
2428                            <1> seektell_err:
2429                            <1> device_rw_err:
2430 0000D19E B80A000000        <1>     mov     eax, ERR_FILE_NOT_OPEN ; file not open !
2431 0000D1A3 A3[A08E0100]      <1>     mov     [u.error], eax ;
2432 0000D1A8 A3[348E0100]      <1>     mov     [u.r0], eax ; ! invalid handle !
2433 0000D1AD E9ACF8FFFF        <1>     jmp     error
2434                            <1>
2435                            <1> sysread: ; < read from file >
2436                            <1>     ; 27/09/2024
2437                            <1>     ; 18/09/2024
2438                            <1>     ; 03/09/2024 (TRDOS v2.0.9)
2439                            <1>     ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
2440                            <1>     ;            -derived from INT_21H.ASM-
2441                            <1>     ;             ("loc_INT21h_read_file")
2442                            <1>         ;   13/03/2011 (05/03/2011)
2443                            <1>         ;   INT 21h Function AH = 3Fh
2444                            <1>         ;   Read from a File
2445                            <1>         ;   INPUT
2446                            <1>     ;          BX = File Handle
2447                            <1>         ;      CX = Number of bytes to read
2448                            <1>         ;           DS:DX= Buffer address
```

```
2449                                  <1>         ;
2450                                  <1>     ; Note: TRDOS 386 'sysread' has been derived from
2451                                  <1>     ;       Retro UNIX 386 v1 'sysread', except a few
2452                                  <1>     ;       code modifications.
2453                                  <1>     ;
2454                                  <1>     ; 13/05/2015 (Retro UNIX 386 v1)
2455                                  <1>     ; 11/05/2015 (Retro UNIX 386 v1 - Beginning)
2456                                  <1>     ; 23/05/2013 (Retro UNIX 8086 v1)
2457                                  <1>     ;
2458                                  <1>     ; 'sysread' is given a buffer to read into and the number of
2459                                  <1>     ; characters to be read. If finds the file from the file
2460                                  <1>     ; descriptor located in *u.r0 (r0). This file descriptor
2461                                  <1>     ; is returned from a successful open call (sysopen).
2462                                  <1>     ; The i-number of file is obtained via 'rw1' and the data
2463                                  <1>     ; is read into core via 'readi'.
2464                                  <1>     ;
2465                                  <1>     ; Calling sequence:
2466                                  <1>     ;       sysread; buffer; nchars
2467                                  <1>     ; Arguments:
2468                                  <1>     ;       buffer - location of contiguous bytes where
2469                                  <1>     ;                input will be placed.
2470                                  <1>     ;       nchars - number of bytes or characters to be read.
2471                                  <1>     ; Inputs: *u.r0 - file descriptor (& arguments)
2472                                  <1>     ; Outputs: *u.r0 - number of bytes read.
2473                                  <1>     ; ..........................................................
2474                                  <1>     ;
2475                                  <1>     ; Retro UNIX 8086 v1 modification:
2476                                  <1>     ;       'sysread' system call has three arguments; so,
2477                                  <1>     ;       * 1st argument, file descriptor is in BX register
2478                                  <1>     ;       * 2nd argument, buffer address/offset in CX register
2479                                  <1>     ;       * 3rd argument, number of bytes is in DX register
2480                                  <1>     ;
2481                                  <1>     ;       AX register (will be restored via 'u.r0') will return
2482                                  <1>     ;       to the user with number of bytes read.
2483                                  <1>     ;
2484                                  <1>     ; TRDOS 386 (05/10/2016)
2485                                  <1>     ;
2486                                  <1>         ; INPUT ->
2487                                  <1>         ;       EBX = File handle (descriptor/index)
2488                                  <1>     ;           ECX = Buffer address
2489                                  <1>         ;           EDX = Number of bytes
2490                                  <1>     ; OUTPUT ->
2491                                  <1>     ;           EAX = Number of bytes have been read
2492                                  <1>     ;           cf = 1 -> Error code in AL
2493                                  <1>     ;
2494                                  <1>     ; Modified Registers: EAX (at the return of system call)
2495                                  <1>     ;
2496                                  <1>
2497                                  <1>     ; EBX = File descriptor
2498 0000D1B2 E8DA300000              <1>     call    getf1
2499 0000D1B7 723A                   <1>     jc      short device_read ; read data from device
2500                                  <1>
2501                                  <1>     ; EAX = First cluster of the file
2502                                  <1>
2503 0000D1B9 E837000000             <1>     call    rw1     ; 03/09/2024 (major modification)
2504 0000D1BE 730A                   <1>     jnc     short sysread_0
2505                                  <1>
2506                                  <1> sysrw_err:  ; 03/09/2024
2507 0000D1C0 A3[348E0100]           <1>     mov     [u.r0], eax ; error code
2508 0000D1C5 E994F8FFFF             <1>     jmp     error
2509                                  <1>
2510                                  <1> sysread_0:
2511 0000D1CA E886340000             <1>     call    readi
2512 0000D1CF EB13                   <1>     jmp     short rw0
2513                                  <1>
2514                                  <1> syswrite: ; < write to file >
2515                                  <1>     ; 27/09/2024
2516                                  <1>     ; 18/09/2024
2517                                  <1>     ; 03/09/2024
2518                                  <1>     ; 25/08/2024 - TRDOS 386 v2.0.9
2519                                  <1>     ; 23/10/2016
2520                                  <1>     ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
2521                                  <1>     ;             -derived from INT_21H.ASM-
2522                                  <1>     ;              ("loc_INT21h_write_file")
2523                                  <1>         ;   13/03/2011 (05/03/2011)
2524                                  <1>         ;   INT 21h Function AH = 40h
2525                                  <1>         ;   Write to a File
2526                                  <1>         ;   INPUT
2527                                  <1>     ;           BX = File Handle
2528                                  <1>         ;       CX = Number of bytes to write
2529                                  <1>         ;            DS:DX= Buffer address
2530                                  <1>         ;
2531                                  <1>     ; Note: TRDOS 386 'sysrwrite' has been derived from
2532                                  <1>     ;       Retro UNIX 386 v1 'syswrite', except a few
2533                                  <1>     ;       code modifications.
2534                                  <1>     ;
2535                                  <1>
2536                                  <1>     ; 13/05/2015 (Retro UNIX 386 v1)
2537                                  <1>     ; 11/05/2015 (Retro UNIX 386 v1 - Beginning)
2538                                  <1>     ; 23/05/2013 (Retro UNIX 8086 v1)
2539                                  <1>     ;
2540                                  <1>     ; 'syswrite' is given a buffer to write onto an output file
2541                                  <1>     ; and the number of characters to write. If finds the file
2542                                  <1>     ; from the file descriptor located in *u.r0 (r0). This file
2543                                  <1>     ; descriptor is returned from a successful open or create call
2544                                  <1>     ; (sysopen or syscreat). The i-number of file is obtained via
2545                                  <1>     ; 'rw1' and buffer is written on the output file via 'write'.
2546                                  <1>     ;
2547                                  <1>     ; Calling sequence:
2548                                  <1>     ;       syswrite; buffer; nchars
2549                                  <1>     ; Arguments:
2550                                  <1>     ;       buffer - location of contiguous bytes to be writtten.
2551                                  <1>     ;       nchars - number of characters to be written.
2552                                  <1>     ; Inputs: *u.r0 - file descriptor (& arguments)
2553                                  <1>     ; Outputs: *u.r0 - number of bytes written.
2554                                  <1>     ; ..........................................................
2555                                  <1>     ;
2556                                  <1>     ; Retro UNIX 8086 v1 modification:
2557                                  <1>     ;       'syswrite' system call has three arguments; so,
2558                                  <1>     ;       * 1st argument, file descriptor is in BX register
2559                                  <1>     ;       * 2nd argument, buffer address/offset in CX register
2560                                  <1>     ;       * 3rd argument, number of bytes is in DX register
2561                                  <1>     ;
2562                                  <1>     ;       AX register (will be restored via 'u.r0') will return
2563                                  <1>     ;       to the user with number of bytes written.
2564                                  <1>     ;
2565                                  <1>     ; INPUT ->
2566                                  <1>         ;       EBX = File handle (descriptor/index)
2567                                  <1>     ;           ECX = Buffer address
2568                                  <1>         ;           EDX = Number of bytes
2569                                  <1>     ; OUTPUT ->
2570                                  <1>     ;           EAX = Number of bytes have been written
2571                                  <1>     ;           cf = 1 -> Error code in AL
2572                                  <1>     ;
```

```
2573                                <1>     ; Modified Registers: EAX (at the return of system call)
2574                                <1>     ;
2575                                <1>
2576                                <1>     ; EBX = File descriptor
2577 0000D1D1 E8BB300000           <1>     call    getf1
2578 0000D1D6 721B                 <1>     jc      short device_write ; write data to device
2579                                <1>     ; EAX = First cluster of the file
2580                                <1>     ; EBX = File number (Open file number) ; 23/10/2016
2581                                <1>
2582 0000D1D8 E818000000           <1>     call    rw1     ; 03/09/2024 (major modification)
2583                                <1>     ; 03/09/2024
2584 0000D1DD 72E1                 <1>     jc      short sysrw_err
2585                                <1>
2586                                <1>     ; 25/08/2024 (*)
2587                                <1>     ;jnc    short syswrite_0
2588                                <1>     ;mov    [u.r0], eax ; error code
2589                                <1>     ;jmp    error
2590                                <1>
2591                                <1>     ; 25/08/2024 - TRDOS 386 v2.0.9 (bugfix) (*)
2592                                <1>     ; (if eax = 0, lets add a cluster at 'mget_w_0')
2593                                <1>     ; ('mget_w' and 'add_new_cluster' procs are modified)
2594                                <1>
2595                                <1> syswrite_0:
2596 0000D1DF E84A3B0000           <1>     call    writei  ; 24/08/2024 ('mget_w' modification)
2597                                <1> rw0: ; 1:
2598 0000D1E4 A1[608E0100]         <1>         mov eax, [u.nread]
2599 0000D1E9 A3[348E0100]         <1>     mov     [u.r0], eax
2600 0000D1EE E98BF8FFFF           <1>     jmp     sysret
2601                                <1>
2602                                <1>     ; 17/04/2021 (temporary)
2603                                <1> device_write:
2604                                <1> device_read:
2605                                <1>     ; 26/09/2024
2606                                <1>     ; 18/09/2024 - TRDOS 386 v2.0.9
2607                                <1>     ; 17/04/2021 - TRDOS 386 v2.0.4
2608                                <1>     ;       (temporary modifications)
2609                                <1>     ;
2610                                <1>     ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
2611                                <1>     ; cl = DEV_OPENMODE ; open mode
2612                                <1>     ; ch = DEV_ACCESS   ; access flags
2613                                <1>     ; al = DEV_DRIVER   ; device number (eax)
2614                                <1>
2615                                <1>     ; 24/04/2025 - TRDOS 386 v2.0.10
2616                                <1>     ;; 18/09/2024 (temporary)
2617                                <1>     ;call  rw2 ; file not open ; cf = 1
2618                                <1>     ;;jmp  error
2619                                <1>     ;; 26/09/2024
2620                                <1>     ;jmp    short sysrw_err
2621                                <1>     ; 24/04/2025
2622 0000D1F3 EBA9                 <1>     jmp     short device_rw_err
2623                                <1>
2624                                <1>
2625                                <1> ;   test    cl, 1 ; 1 = read, 2 = write, 3 = read&write
2626                                <1> ;   jz      short rw3
2627                                <1> ;
2628                                <1> ;   mov     ebx, eax
2629                                <1> ;   shl     bx, 2 ; *4
2630                                <1> ;
2631                                <1> ;   test    ch, 80h ; bit 7, installable device driver flag
2632                                <1> ;   jz      short d_read_2 ; Kernel device
2633                                <1> ;   ; installable device
2634                                <1> ;d_read_1:
2635                                <1> ;       jmp dword [ebx+IDEV_RADDR-4]
2636                                <1> ;d_read_2:
2637                                <1> ;   jmp     dword [ebx+KDEV_RADDR-4]
2638                                <1>
2639                                <1> ;device_write:
2640                                <1>     ; 17/04/2021 - TRDOS 386 v2.0.4
2641                                <1>     ;       (temporary modifications)
2642                                <1>     ;
2643                                <1>     ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
2644                                <1>     ; cl = DEV_OPENMODE ; open mode
2645                                <1>     ; ch = DEV_ACCESS   ; access flags
2646                                <1>     ; al = DEV_DRIVER   ; device number (eax)
2647                                <1>
2648                                <1>     ; 17/04/2021 (temporary)
2649                                <1>     ;jmp    short rw2 ; file not open
2650                                <1>
2651                                <1> ;   test    cl, 2 ; 1 = read, 2 = write, 3 = read&write
2652                                <1> ;   jz      short rw3
2653                                <1> ;
2654                                <1> ;   mov     ebx, eax
2655                                <1> ;   shl     bx, 2 ; *4
2656                                <1> ;
2657                                <1> ;   test    ch, 80h ; bit 7, installable device driver flag
2658                                <1> ;   jz      short d_write_2 ; Kernel device
2659                                <1> ;   ; installable device
2660                                <1> ;d_write_1:
2661                                <1> ;       jmp dword [ebx+IDEV_WADDR-4]
2662                                <1> ;d_write_2:
2663                                <1> ;   jmp     dword [ebx+KDEV_WADDR-4]
2664                                <1>
2665                                <1> rw1:
2666                                <1>     ; 27/09/2024
2667                                <1>     ; 03/09/2024 (TRDOS 386 v2.0.9)
2668                                <1>     ; 17/04/2021 (TRDOS 386 v2.0.4)
2669                                <1>     ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
2670                                <1>     ; 14/05/2015 (Retro UNIX 386 v1)
2671                                <1>     ; 11/05/2015 (Retro UNIX 386 v1 - Beginning)
2672                                <1>     ; 23/05/2013 - 24/05/2013 (Retro UNIX 8086 v1)
2673                                <1>     ; System call registers: ebx, ecx, edx (through 'sysenter')
2674                                <1>     ;
2675                                <1>     ; EBX = File descriptor
2676                                <1>     ;call  getf1 ; calling point in 'getf' from 'rw1'
2677                                <1>     ;jc     short device_rw ; read/write data from/to device
2678                                <1>     ; EAX = First cluster of the file
2679                                <1>
2680                                <1>     ; 03/09/2024
2681 0000D1F5 09C0                 <1>     or      eax, eax
2682 0000D1F7 7421                 <1>     jz      short rw7 ; eax = 0 -> empty file (OK for now)
2683                                <1>
2684 0000D1F9 83F802               <1>     cmp     eax, 2    ; is it valid cluster number ?
2685                                <1>     ;jb     short rw2
2686                                <1>     ; 03/09/2024
2687 0000D1FC 7307                 <1>     jnb     short rw6 ; yes, check upper limit
2688                                <1>
2689                                <1>     ;;;
2690                                <1>     ; eax = 1 -> invalid cluster number
2691                                <1> rw5:
2692 0000D1FE B823000000           <1>     mov     eax, ERR_CLUSTER ; 35 ; 'cluster not available !'
2693 0000D203 EB2C                 <1>     jmp     short rw4 ; cf = 1
2694                                <1>
2695                                <1> rw6:; 03/09/2024 (check cluster number is valid or not)
2696 0000D205 53                   <1>     push    ebx
```

```
2697                                <1>     ; ebx <= OPENFILES-1 ; 0-31
2698 0000D206 8ABB[C4830100]        <1>     mov     bh, [ebx+OF_DRIVE] ; drive number * 256
2699 0000D20C 28DB                  <1>     sub     bl, bl ; 27/09/2024
2700 0000D20E 8B9B78010900          <1>     mov     ebx, [ebx+Logical_DOSDisks+LD_Clusters]
2701 0000D214 43                    <1>     inc     ebx ; cluster count + 1 = last cluster number
2702 0000D215 39C3                  <1>     cmp     ebx, eax
2703 0000D217 5B                    <1>     pop     ebx
2704 0000D218 72E4                  <1>     jb      short rw5
2705                                <1> rw7:
2706                                <1>     ;;;
2707                                <1>
2708 0000D21A 890D[588E0100]        <1>     mov     [u.base], ecx  ; buffer address/offset
2709                                <1>                            ;(in the user's virtual memory space)
2710 0000D220 8915[5C8E0100]        <1>     mov     [u.count], edx
2711 0000D226 C705[A08E0100]0000-   <1>         mov dword [u.error], 0 ; reset the last error code
2711 0000D22E 0000                  <1>
2712 0000D230 C3                    <1>     retn
2713                                <1>
2714                                <1> ; 24/04/2025 - TRDOS 386 v2.0.10
2715                                <1> %if 0
2716                                <1>
2717                                <1> rw2:
2718                                <1>     mov     eax, ERR_FILE_NOT_OPEN ; file not open !
2719                                <1>     ;mov    dword [u.error], eax
2720                                <1>     ;retn
2721                                <1>
2722                                <1> %endif
2723                                <1>     ; 03/09/2024
2724                                <1>     ; 17/04/2021
2725                                <1>     ;jmp    short rw4
2726                                <1>
2727                                <1>     ; 03/09/2024
2728                                <1> rw3:
2729                                <1> ;   mov     eax, ERR_FILE_ACCESS ; permission denied !
2730                                <1> ;   stc
2731                                <1>
2732                                <1> rw4:; 17/04/2021
2733 0000D231 A3[A08E0100]          <1>     mov     dword [u.error], eax
2734 0000D236 C3                    <1>     retn
2735                                <1>
2736                                <1> systimer:
2737                                <1>     ; 23/07/2022 - TRDOS 386 Kernel v2.0.5
2738                                <1>     ; 02/01/2017
2739                                <1>     ; 21/12/2016
2740                                <1>     ; 19/12/2016
2741                                <1>     ; 10/12/2016 (callback)
2742                                <1>     ; 10/06/2016
2743                                <1>     ; 07/06/2016
2744                                <1>     ; 06/06/2016
2745                                <1>     ; 21/05/2016
2746                                <1>     ; 19/05/2016
2747                                <1>     ; 18/05/2016 - TRDOS 386 (TRDOS v2.0)
2748                                <1>     ; (TRDOS 386 feature only!)
2749                                <1>     ;
2750                                <1>     ; (start or stop timer event(s))
2751                                <1>     ;
2752                                <1>     ; INPUT ->
2753                                <1>     ;       BL = Signal return byte (response byte)
2754                                <1>     ;            (Any requested value between 0 and 255)
2755                                <1>     ;            (Kernel will put it at the requested address)
2756                                <1>     ;       BH = Time count unit
2757                                <1>     ;            0 = Stop timer event
2758                                <1>     ;            1 = 18.2 ticks per second
2759                                <1>     ;            2 = 10 milliseconds
2760                                <1>     ;            3 = 1 second (for real time clock interrupt)
2761                                <1>     ;            4 = time/tick count in current time count unit
2762                                <1>     ;           // 10/12/2016
2763                                <1>     ;           80h = Stop timer event (callback method)
2764                                <1>     ;           81h = 18.2 ticks per second, callback method
2765                                <1>     ;           82h = 10 milliseconds, callback method
2766                                <1>     ;           83h = 1 second (for RTC int), callback method
2767                                <1>     ;           84h = current time count unit, callback method
2768                                <1>     ;
2769                                <1>     ;           Note: Only 03h or 83h will set real time clock
2770                                <1>     ;                (RTC) events (Others are for PIT events)!
2771                                <1>     ;
2772                                <1>     ;       NOTE: If callback (user service) method is used,
2773                                <1>     ;           EDX will point to the return address (of service
2774                                <1>     ;           procedure) in user's space instead of signal
2775                                <1>     ;           response byte address. (TRDOS 386 kernel will
2776                                <1>     ;           direct the cpu to that address -in user's space-
2777                                <1>     ;           at the return of system call or interrupt
2778                                <1>     ;           just after the adjusted count/time is elapsed.)
2779                                <1>     ;           User's sevice routine must be ended with a
2780                                <1>     ;           'iret'. Normal return addresses from system
2781                                <1>     ;           calls or and interrupts will be kept same except
2782                                <1>     ;           the timer returns.
2783                                <1>     ;
2784                                <1>     ;       BH = 0 -> Stop timer event
2785                                <1>     ;       BL = Timer event number (1 to 255) if BH = 0
2786                                <1>     ;            If BL = 0, all timer events (which are belongs
2787                                <1>     ;             to running process) will be stopped
2788                                <1>     ;       ECX = Time/Tick count (depending on time count unit)
2789                                <1>     ;       EDX = Signal return (Response) byte address
2790                                <1>     ;             (virtual address in user's memory space)
2791                                <1>     ; OUTPUT ->
2792                                <1>     ;       AL = Timer event number      (1 to 255) (max. value = 16)
2793                                <1>     ;       IF BH Input = 0 & CF = 0 & AL = 0 ->
2794                                <1>     ;            timer event(s) has/have been stopped/finished
2795                                <1>     ;       CF = 1 & AL = 0 -> no timer setting space to set
2796                                <1>     ;       CF = 1 & AL > 0 -> timer count unit is not usable
2797                                <1>     ;
2798                                <1>     ;       NOTE: To modify a time count for a user function,
2799                                <1>     ;             at first, current timer event must be stopped
2800                                <1>     ;             then a new timer event (which is related with
2801                                <1>     ;             same user function) must be started.
2802                                <1>     ;
2803                                <1>     ;             Signal return (response) byte may be used for
2804                                <1>     ;             several purposes. Kernel will put this value
2805                                <1>     ;             to requested address during timer interrupt,
2806                                <1>     ;             program/user can check this value to understand
2807                                <1>     ;             which event has been occurred and what is changed.
2808                                <1>     ;             (Multi timer events can share same signal address)
2809                                <1>     ;
2810                                <1>     ;       NOTE: If the process is running while the time count
2811                                <1>     ;             is reached, kernel will put signal return (response)
2812                                <1>     ;             byte value at requested address during timer
2813                                <1>     ;             interrupt and the process will continue to run.
2814                                <1>     ;             Program/process must call (jump to) it's timer event
2815                                <1>     ;             function as required, for checking the timer event
2816                                <1>     ;             status via signal return (response) byte address.
2817                                <1>     ;
2818                                <1>     ;             If the process is not running (waiting or sleeping
2819                                <1>     ;             or released) while the time count is reached,
```

```
2820                                    <1>     ;           it is restarted from where it left, to ensure
2821                                    <1>     ;           proper multi media (video, audio, clock, timer)
2822                                    <1>     ;           functionality.
2823                                    <1>     ;
2824                                    <1>     ;           (It is better to use 'syswait' or 'syssleep',
2825                                    <1>     ;           or 'sysrele' system call just after the timer
2826                                    <1>     ;           function. Otherwise, timer events may block other
2827                                    <1>     ;           processes which are not using timer events.)
2828                                    <1>     ;
2829                                    <1>     ; Timer Event Structure: (max. 16 timer events, 16*16 bytes)
2830                                    <1>     ;       Owner:         resb 1 ; 0 = free
2831                                    <1>     ;                             ;>0 = process number (u.uno)
2832                                    <1>     ;       Calback:       resb 1 ; 1 = callback, 0 = response byte
2833                                    <1>     ;       Interrupt:      resb 1 ; 0 = Timer interrupt (or none)
2834                                    <1>     ;                             ; 1 = Real Time Clock interrupt
2835                                    <1>     ;       Response:       resb 1 ; 0 to 255, signal return value
2836                                    <1>     ;       Count Limit:   resd 1 ; count of ticks (total/set)
2837                                    <1>     ;       Current Count:        resd 1 ; count of ticks (current)
2838                                    <1>     ;       Response Addr:  resd 1 ; response byte (pointer) address
2839                                    <1>     ;
2840                                    <1>
2841                                    <1>     ; 19/12/2016 (timer callback)
2842 0000D237 C605[1C880100]00         <1>     mov     byte [tcallback], 0
2843 0000D23E C605[1D880100]00         <1>     mov     byte [trtc], 0
2844 0000D245 C705[A88E0100]0000-      <1>     mov     dword [u.tcb], 0 ; this is not necessary...
2844 0000D24D 0000                     <1>
2845                                    <1>
2846 0000D24F 80FF80                   <1>     cmp     bh, 80h
2847 0000D252 7224                     <1>     jb      short systimer_cb2
2848 0000D254 7704                     <1>     ja      short systimer_cb0
2849                                    <1>
2850 0000D256 31D2                     <1>     xor     edx, edx ; 0, reset callback address
2851 0000D258 EB0B                     <1>     jmp     short systimer_cb1
2852                                    <1>
2853                                    <1> systimer_cb0:
2854 0000D25A 80FF84                   <1>     cmp     bh, 84h
2855 0000D25D 7764                     <1>     ja      short systimer_5 ;  undefined, error
2856                                    <1>
2857                                    <1>     ;mov    byte [tcallback], 1 ; 19/12/2016
2858 0000D25F FE05[1C880100]           <1>     inc     byte [tcallback]
2859                                    <1>
2860                                    <1> systimer_cb1:
2861 0000D265 0FB635[858E0100]         <1>     movzx   esi, byte [u.uno] ; process number
2862                                    <1>     ;shl    si, 2
2863                                    <1>     ; 23/07/2022
2864 0000D26C C1E602                   <1>     shl     esi, 2
2865 0000D26F 8996[C48D0100]           <1>     mov     [esi+p.tcb-4], edx ; set process timer callback address
2866                                    <1>                                ; (overwrite prev value if it is set!)
2867 0000D275 80E77F                   <1>     and     bh, 7Fh
2868                                    <1>
2869                                    <1> systimer_cb2:
2870 0000D278 80FF02                   <1>     cmp     bh, 2
2871 0000D27B 7446                     <1>         je      short systimer_5  ; only 18.2 ticks per second is usable
2872                                    <1>                               ; 10 milliseconds (100 Hertz) timer
2873                                    <1>                               ; will be set later (18/05/2016)
2874 0000D27D 774C                     <1>         ja      short systimer_6
2875                                    <1>
2876 0000D27F 20FF                     <1>     and     bh, bh
2877                                    <1>     ;jz     systimer_9        ; stop timer event(s)
2878                                    <1>     ; 23/07/2022
2879 0000D281 7505                     <1>     jnz     short systimer_19
2880 0000D283 E9BA000000               <1>     jmp     systimer_9
2881                                    <1>
2882                                    <1>     ; bh = 1 (timer interrupt, 18.2 Hz, IBM PC/AT ROMBIOS default)
2883                                    <1>
2884                                    <1> systimer_19:
2885 0000D288 B00A                     <1>     mov     al, 10 ; (*)
2886                                    <1>
2887                                    <1> systimer_0:
2888 0000D28A B710                     <1>     mov     bh, 16
2889                                    <1>     ;
2890 0000D28C 383D[13830100]           <1>     cmp     [timer_events], bh ; 16 ; 07/06/2016
2891 0000D292 7319                     <1>     jnb     short systimer_3  ; max. 16 timer events
2892                                    <1>     ;
2893 0000D294 50                       <1>     push    eax ; (*)
2894                                    <1>
2895 0000D295 BF[488F0100]             <1>     mov     edi, timer_set  ; beginning address of timer events
2896                                    <1>                             ; setting space
2897 0000D29A 30C0                     <1>     xor     al, al ; 0
2898                                    <1> systimer_1:
2899 0000D29C FEC0                     <1>     inc     al
2900 0000D29E 803F00                   <1>     cmp     byte [edi], 0  ; is it free space ?
2901 0000D2A1 7639                     <1>     jna     short systimer_7 ; yes
2902 0000D2A3 FECF                     <1>     dec     bh
2903 0000D2A5 7405                     <1>     jz      short systimer_2
2904 0000D2A7 83C710                   <1>     add     edi, 16
2905 0000D2AA EBF0                     <1>     jmp     short  systimer_1 ; next event space
2906                                    <1>
2907                                    <1> systimer_2:
2908 0000D2AC 58                       <1>     pop     eax ; (*) discard
2909                                    <1> systimer_3:
2910 0000D2AD C605[348E0100]00         <1>     mov     byte [u.r0], 0
2911                                    <1> systimer_4:
2912 0000D2B4 C705[A08E0100]1B00-      <1>         mov     dword [u.error], ERR_MISC
2912 0000D2BC 0000                     <1>
2913                                    <1>                                     ; one of miscellaneous/other errors
2914 0000D2BE E99BF7FFFF               <1>     jmp     error ; cf -> 1
2915                                    <1>
2916                                    <1> systimer_5:
2917 0000D2C3 883D[348E0100]           <1>     mov     [u.r0], bh ; Time count unit (=2 or >3)
2918 0000D2C9 EBE9                     <1>     jmp     short systimer_4 ; 07/06/2016
2919                                    <1>
2920                                    <1> systimer_6:
2921 0000D2CB 80FF04                   <1>     cmp     bh, 4
2922 0000D2CE 77F3                     <1>         ja      short systimer_5  ; undefined time count unit
2923                                    <1>     ;jb     short systimer_16
2924                                    <1>
2925                                    <1>     ;mov    al, 1   ; default (use current timer unit)
2926                                    <1>                     ; countdown value is in ECX !
2927                                    <1>                     ; max. value of ecx = 4294967296/10
2928                                    <1>         ;jmp    short systimer_0
2929                                    <1>     ;jmp    short systimer_19
2930 0000D2D0 74B6                     <1>     je      short systimer_19
2931                                    <1>
2932                                    <1> systimer_16:
2933                                    <1>     ; bh = 3
2934                                    <1>     ; timer event via real time clock interrupt
2935                                    <1>     ; interrupt/update frequency: 1 Hz (1 tick per second)
2936                                    <1>
2937 0000D2D2 B0B6                     <1>     mov     al, 182 ; (*) ; 18.2 * 10
2938 0000D2D4 FE05[1D880100]           <1>     inc     byte [trtc] ; timer event via real time clock
2939 0000D2DA EBAE                     <1>         jmp     short systimer_0
2940                                    <1>
2941                                    <1> systimer_7:
```

```
2942 0000D2DC A2[348E0100]        <1>     mov     [u.r0], al ; timer event number
2943                              <1>     ;
2944                              <1>     ; edi = address of empty timer event area
2945 0000D2E1 A0[858E0100]        <1>     mov     al, [u.uno]
2946 0000D2E6 FA                  <1>     cli     ; disable interrupts
2947 0000D2E7 AA                  <1>     stosb   ; process number
2948 0000D2E8 A0[1C880100]        <1>     mov     al, [tcallback] ; timer callback flag
2949 0000D2ED AA                  <1>     stosb   ; 1= callback method, 0= signal response byte method
2950 0000D2EE A0[1D880100]        <1>     mov     al, [trtc] ; timer interrupt type
2951 0000D2F3 AA                  <1>     stosb   ; 1= real time clock, 0= programmable interval timer
2952 0000D2F4 88D8                <1>     mov     al, bl ; Signal return (Response) value
2953 0000D2F6 AA                  <1>     stosb    ; response byte
2954 0000D2F7 58                  <1>     pop     eax ; (*) ; 10 or 182
2955 0000D2F8 89D3                <1>     mov     ebx, edx ; virtual address for response/signal byte
2956 0000D2FA F7E1                <1>     mul     ecx
2957                              <1>     ; (eax = 10 * count of 18.2 Hz timer ticks)
2958                              <1>     ; (count down step = 10)
2959 0000D2FC AB                  <1>     stosd   ; count limit (reset value)
2960 0000D2FD AB                  <1>     stosd   ; current count value
2961                              <1>
2962                              <1>     ; 19/12/2016
2963 0000D2FE 803D[1C880100]00    <1>     cmp     byte [tcallback], 0 ; timer callback method ?
2964 0000D305 7604                <1>     jna     short systimer_17 ; no
2965 0000D307 89D8                <1>     mov     eax, ebx ; virtual address for callback routine
2966 0000D309 EB0D                <1>     jmp     short systimer_18
2967                              <1>
2968                              <1> systimer_17: ; signal response byte method
2969                              <1>     ; ebx = virtual address
2970                              <1>     ; [u.pgdir] = page directory's physical address
2971                              <1>     ; 20/02/2017
2972 0000D30B FE05[1E880100]      <1>     inc     byte [no_page_swap] ; 1
2973                              <1>                     ; Do not add this page to swap queue
2974                              <1>                     ; and remove it from swap queue if it is
2975                              <1>                     ; on the queue.
2976 0000D311 E8EE88FFFF          <1>     call    get_physical_addr
2977 0000D316 721A                <1>     jc      short systimer_8 ; 07/06/2016
2978                              <1>     ; eax = physical address of the virtual address in user's space
2979                              <1> systimer_18:
2980 0000D318 AB                  <1>     stosd   ; response addr (physical) or callback addr (virtual)
2981 0000D319 FE05[13830100]      <1>     inc     byte [timer_events] ; 07/06/201
2982                              <1>     ; 02/01/2017
2983 0000D31F 0FB605[858E0100]    <1>     movzx   eax, byte [u.uno]
2984 0000D326 FE80[B78D0100]      <1>     inc     byte [eax+p.timer-1]
2985                              <1>     ;
2986 0000D32C FB                  <1>     sti     ; enable interrupts
2987 0000D32D E94CF7FFFF          <1>     jmp     sysret
2988                              <1>
2989                              <1> systimer_8:
2990                              <1>     ; 10/06/2016
2991                              <1>     ; 07/06/2016
2992 0000D332 28C0                <1>     sub     al, al ; 0
2993 0000D334 8847F4              <1>     mov     [edi-12], al ; clear process number (free timer event)
2994                              <1>     ;mov    dword [edi], eax ; 0
2995 0000D337 FB                  <1>     sti
2996 0000D338 A2[348E0100]        <1>     mov     [u.r0], al ; 0
2997 0000D33D E91CF7FFFF          <1>     jmp     error
2998                              <1>
2999                              <1> systimer_9:
3000                              <1>     ; 10/06/2016
3001                              <1>     ; 07/06/2016
3002 0000D342 28C0                <1>     sub     al, al
3003 0000D344 A2[348E0100]        <1>     mov     byte [u.r0], al ; 0
3004 0000D349 3805[13830100]      <1>     cmp     byte [timer_events], al ;  0
3005 0000D34F 7631                <1>     jna     short systimer_12
3006                              <1>
3007                              <1>     ; Note: ecx and edx are undefined here
3008                              <1>     ;       (for stop timer function)
3009                              <1>
3010 0000D351 BE[488F0100]        <1>     mov     esi, timer_set  ; beginning address of timer events
3011                              <1>                             ; setting space
3012 0000D356 A0[858E0100]        <1>     mov     al, [u.uno]
3013                              <1>
3014 0000D35B B710                <1>     mov     bh, 16
3015                              <1>
3016 0000D35D 08DB                <1>     or      bl, bl
3017 0000D35F 7544                <1>     jnz     short systimer_15
3018                              <1>
3019                              <1>     ; clear timer event areas belong to current process
3020                              <1>     ; (for stopping all timer events belong to current process)
3021 0000D361 FA                  <1>     cli     ; disable interrupts
3022                              <1> systimer_10:
3023                              <1>     ; 10/06/2016
3024                              <1>     ; 07/06/2016
3025 0000D362 8A26                <1>     mov     ah, [esi]
3026 0000D364 08E4                <1>     or      ah, ah ; 0 ?
3027 0000D366 7411                <1>     jz      short systimer_11
3028 0000D368 38C4                <1>     cmp     ah, al ; is the process number (owner) same ?
3029 0000D36A 750D                <1>         jne     short systimer_11 ; no
3030                              <1>
3031                              <1>     ;mov    byte [esi], 0
3032 0000D36C 66C7060000          <1>     mov     word [esi], 0 ; clear
3033                              <1>     ;mov    dword [esi+12], 0 ; clear
3034                              <1>
3035 0000D371 FE0D[13830100]      <1>     dec     byte [timer_events]
3036 0000D377 7409                <1>     jz      short systimer_12
3037                              <1>
3038                              <1> systimer_11:
3039 0000D379 FECF                <1>     dec     bh
3040 0000D37B 7405                <1>     jz      short systimer_12
3041 0000D37D 83C610              <1>     add     esi, 16
3042 0000D380 EBE0                <1>     jmp     short systimer_10
3043                              <1>
3044                              <1> systimer_12:
3045 0000D382 0FB635[858E0100]    <1>     movzx   esi, byte [u.uno]
3046 0000D389 08DB                <1>     or      bl, bl ; all timer events or one timer event ?
3047 0000D38B 740C                <1>     jz      short systimer_13
3048 0000D38D 8A9E[B78D0100]      <1>     mov     bl, [esi+p.timer-1]
3049 0000D393 20DB                <1>     and     bl, bl  ; previous number of timer events for the process
3050 0000D395 7408                <1>     jz      short systimer_14
3051 0000D397 FECB                <1>     dec     bl  ; previous number of timer events for the process - 1
3052                              <1> systimer_13:
3053 0000D399 889E[B78D0100]      <1>     mov     [esi+p.timer-1], bl ; 0 ; no timer events for process
3054                              <1> systimer_14:
3055 0000D39F FB                  <1>     sti     ; enable interrupts
3056 0000D3A0 E9D9F6FFFF          <1>     jmp     sysret
3057                              <1>
3058                              <1> systimer_15:
3059 0000D3A5 38FB                <1>     cmp     bl, bh ; 16
3060                              <1>         ;ja systimer_4      ; max. 16 timer events !
3061                              <1>     ; 23/07/2022
3062 0000D3A7 7605                <1>     jna     short systimer_21
3063                              <1> systimer_20:
3064 0000D3A9 E906FFFFFF          <1>     jmp     systimer_4
3065                              <1> systimer_21:        ; 23/07/2022
```

```
0000D3AE 88DA                <1>     mov     dl, bl
0000D3B0 FECA                <1>     dec     dl  ; 16 -> 15 ... 1 -> 0
0000D3B2 C0E204              <1>     shl     dl, 4 ; * 16
0000D3B5 0FB6FA              <1>     movzx   edi, dl
0000D3B8 01F7                <1>     add     edi, esi ; timer_set
                             <1>
0000D3BA 3A07                <1>     cmp     al, [edi] ; process number
                             <1>         ;jne       systimer_4
                             <1>     ; 23/07/2022
0000D3BC 75EB                <1>     jne     short systimer_20 ; jmp systimer_4
                             <1>
                             <1>     ; same process ID
0000D3BE FA                  <1>     cli     ; disable interrupts
                             <1>     ; 10/06/2016 ; 02/01/2017
                             <1>     ;mov    byte [edi], 0
0000D3BF 66C7070000          <1>     mov     word [edi], 0 ; clear
                             <1>     ;mov    dword [edi+12], 0 ; clear
0000D3C4 FE0D[13830100]      <1>     dec     byte [timer_events]
0000D3CA EBB6                <1>     jmp     short systimer_12
                             <1>
                             <1> sysvideo: ; VIDEO DATA TRANSFER FUNCTIONS
                             <1>     ; 11/08/2022
                             <1>     ; 08/08/2022
                             <1>     ; 23/07/2022 (TRDOS 386 v2.0.5)
                             <1>     ; 06/03/2021
                             <1>     ; 02/03/2021
                             <1>     ; 28/02/2021
                             <1>     ; 27/02/2021
                             <1>     ; 26/02/2021
                             <1>     ; 25/02/2021
                             <1>     ; 21/02/2021, 22/02/2021, 23/02/2021
                             <1>     ; 15/02/2021, 16/02/2021, 18/02/2021
                             <1>     ; 10/02/2021, 11/02/2021, 12/02/2021
                             <1>     ; 07/02/2021, 08/02/2021
                             <1>     ; 01/02/2021, 02/02/2021, 05/02/2021
                             <1>     ; 29/01/2021, 30/01/2021, 31/01/2021
                             <1>     ; 23/01/2021, 24/01/2021, 28/01/2021
                             <1>     ; 18/01/2021, 19/01/2021, 22/01/2021
                             <1>     ; 04/01/2021, 10/01/2021, 11/01/2021
                             <1>     ; 01/01/2021, 02/01/2021, 03/01/2021
                             <1>     ; 28/12/2020, 29/12/2020, 30/12/2020
                             <1>     ; 25/12/2020, 26/12/2020
                             <1>     ; 21/12/2020, 23/12/2020
                             <1>     ; 12/12/2020, 14/12/2020
                             <1>     ; 10/12/2020, 11/12/2020
                             <1>     ; 03/12/2020, 04/12/2020
                             <1>     ; 22/11/2020, 23/11/2020
                             <1>     ; 21/11/2020 (TRDOS 386 v2.0.3)
                             <1>     ; 12/05/2017
                             <1>     ; 11/07/2016
                             <1>     ; 13/06/2016
                             <1>     ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
                             <1>     ;
                             <1>     ; VIDEO DATA TRANSFER FUNCTIONS:
                             <1>     ;
                             <1>     ; Inputs:
                             <1>     ;               ; 07/02/2021
                             <1>     ;       BH = 0 = VIDEO BIOS Mode 3, tty/text mode data transfers
                             <1>     ;            BL =
                             <1>     ;              Bits 0&1, Transfer direction
                             <1>     ;                     0 - System to system
                             <1>     ;                     1 - User to system
                             <1>     ;                     2 - System to user
                             <1>     ;                     3 - Exhange (Swap) - 28/01/2021
                             <1>     ;              Bits 2, Transfer Type
                             <1>     ;                     0 - Display page (complete) transfer
                             <1>     ;                     1 - Display page window (col,row) transfer
                             <1>     ;              ; 28/01/2021
                             <1>     ;              Bits 3..7 - Reserved, undefined (must be 0)
                             <1>     ;               ; 28/01/2021
                             <1>     ;            /// BL = 0 -> System to system (display page) transfer
                             <1>     ;                CL = Source page (0FFh = current video page)
                             <1>     ;                DL = Destination page (0FFh = current video page)
                             <1>     ;                (Note: Nothing to do if src & dest are same page)
                             <1>     ;            /// BL = 1&2 -> user to system & system to user transfer
                             <1>     ;                ECX = User's buffer address
                             <1>     ;                DL = Video page (0FFh = current video page)
                             <1>     ;            /// BL = 3 -> exchange (swap) display page ; 28/01/2021
                             <1>     ;                ECX = User's buffer address
                             <1>     ;                DL = Video page (0FFh = current video page)
                             <1>     ;                EDI = Swap address in user's memory (must be > 0)
                             <1>     ;            /// BL = 5&6&7 -> 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 or BL bit 0 & bit 1 are 1 ; 28/01/2021
                             <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) addr
                             <1>     ;
                             <1>     ;               ; 06/02/2021
                             <1>     ;               ; 05/02/2021
                             <1>     ;               ; 01/02/2021, 02/02/2021
                             <1>     ;               ; 30/01/2021, 31/02/2021
                             <1>     ;               ; 29/01/2021 (major modification)
                             <1>     ;               ; 23/11/2020 (major modification)
                             <1>     ;               ; 22/11/2020 (bugfixes and extensions)
                             <1>     ;       BH = 1 = VGA Graphics (0A0000h) data transfers
                             <1>     ;              BL bit 7
                             <1>     ;                 resolution (screen width) option
                             <1>     ;                 0 = 320 pixels
                             <1>     ;                 1 = 640 pixels
                             <1>     ;              .. followings are same with SVGA transfer function
                             <1>     ;              BL bit 6
                             <1>     ;                 direction option
                             <1>     ;                 0 = user to system (video memory)
                             <1>     ;                 1 = system to user
                             <1>     ;              BL bit 5
                             <1>     ;                 masked/direct (non-masked) operations
                             <1>     ;                 1 = masked, 0 = non-masked (direct)
                             <1>     ;              BL bit 4
                             <1>     ;                 page/window option
                             <1>     ;                 1 = window, 0 = display page (screen)
                             <1>     ;              BL bit 0 to 3 (pixel operation types)
```

```
3190                                    <1>     ;               0 = Copy pixels (colors) ((mask color))
3191                                    <1>     ;               1 = Change (New, Fill) color
3192                                    <1>     ;               2 = Add color
3193                                    <1>     ;               3 = Sub color
3194                                    <1>     ;               4 = OR color
3195                                    <1>     ;               5 = AND color
3196                                    <1>     ;               6 = XOR color
3197                                    <1>     ;               7 = NOT color
3198                                    <1>     ;               8 = NEG color
3199                                    <1>     ;               9 = INC color
3200                                    <1>     ;              10 = DEC color
3201                                    <1>     ;              11 = Mix (Average) colors
3202                                    <1>     ;              12 = Replace pixel colors
3203                                    <1>     ;              13 = Copy pixel block(s)
3204                                    <1>     ;              14 = Write line(s)
3205                                    <1>     ;              15 = Write character (font)
3206                                    <1>     ;
3207                                    <1>     ;          Input Registers for pixel operations:
3208                                    <1>     ;               Same with LFB data transfer function below
3209                                    <1>     ;
3210                                    <1>     ;              ; 25/02/2021
3211                                    <1>     ;              ; 05/02/2021, 06/02/2021
3212                                    <1>     ;              ; 01/02/2021, 02/02/2021
3213                                    <1>     ;              ; 30/01/2021, 31/02/2021
3214                                    <1>     ;              ; 29/01/2021 (major modification)
3215                                    <1>     ;              ; 23/11/2020 (major modification)
3216                                    <1>     ;              ; 22/11/2020 (bugfixes and extensions)
3217                                    <1>     ;       BH = 2 = Super VGA, LINEAR FRAME BUFFER data transfers
3218                                    <1>     ;              BL bit 7
3219                                    <1>     ;                 unused (invalid), must be 0
3220                                    <1>     ;              BL bit 6
3221                                    <1>     ;                 direction option
3222                                    <1>     ;                 0 = user to system (video memory)
3223                                    <1>     ;                 1 = system to user
3224                                    <1>     ;              BL bit 5
3225                                    <1>     ;                 masked/direct (non-masked) operations
3226                                    <1>     ;                 1 = masked, 0 = non-masked (direct)
3227                                    <1>     ;              BL bit 4
3228                                    <1>     ;                 page/window option
3229                                    <1>     ;                 1 = window, 0 = display page (screen)
3230                                    <1>     ;              BL bit 0 to 3 (pixel operation types)
3231                                    <1>     ;                 0 = Copy pixels (colors) ((mask color))
3232                                    <1>     ;                 1 = Change (New, Fill) color
3233                                    <1>     ;                 2 = Add color
3234                                    <1>     ;                 3 = Sub color
3235                                    <1>     ;                 4 = OR color
3236                                    <1>     ;                 5 = AND color
3237                                    <1>     ;                 6 = XOR color
3238                                    <1>     ;                 7 = NOT color
3239                                    <1>     ;                 8 = NEG color
3240                                    <1>     ;                 9 = INC color
3241                                    <1>     ;                10 = DEC color
3242                                    <1>     ;                11 = Mix (Average) colors
3243                                    <1>     ;                12 = Replace pixel colors
3244                                    <1>     ;                13 = Copy pixel block(s)
3245                                    <1>     ;                14 = Write line(s)
3246                                    <1>     ;                15 = Write character (font)
3247                                    <1>     ;
3248                                    <1>     ;          Note: If HW of EBX > 0, it is VESA VBE mode number
3249                                    <1>     ;               otherwise, function will be applied
3250                                    <1>     ;               to current (VESA VBE) video mode.
3251                                    <1>     ;
3252                                    <1>     ;          Input Registers for pixel operations:
3253                                    <1>     ;              -- user to system & system to system --
3254                                    <1>     ;              -- (BL = 0 to 0Fh) -- non-masked, screen --
3255                                    <1>     ;              -- (BL = 10h to 1Fh) -- non-masked, window --
3256                                    <1>     ;              -- (BL = 20h to 2Fh) -- masked, screen --
3257                                    <1>     ;              -- (BL = 30h to 3Fh) -- masked, window --
3258                                    <1>     ;              (*) window, (**) masked (***) sys to sys
3259                                    <1>     ;              for BL bit 0 to 3
3260                                    <1>     ;              00h: COPY PIXELS
3261                                    <1>     ;                If BL bit 4 = 0 ; 21/02/2021
3262                                    <1>     ;                   full screen copy
3263                                    <1>     ;                   ECX & EDX will not be used
3264                                    <1>     ;                  (user buffer must fit to display page)
3265                                    <1>     ;                If BL bit 4 = 1 ; 21/02/2021
3266                                    <1>     ;                  ECX = start position (row, column) (*)
3267                                    <1>     ;                        (HW = row, CX = column)
3268                                    <1>     ;                  EDX = size (rows, colums) (*)
3269                                    <1>     ;                        (HW = rows, DX = columns)
3270                                    <1>     ;                         (0 -> invalid)
3271                                    <1>     ;                         (1 -> horizontal or vertical line)
3272                                    <1>     ;                  ESI = user's buffer address
3273                                    <1>     ;                  EDI = mask color (**) ; 25/02/2021
3274                                    <1>     ;                        (this color will be excluded)
3275                                    <1>     ;              01h: CHANGE PIXEL COLORS
3276                                    <1>     ;              02h: ADD PIXEL COLORS
3277                                    <1>     ;              03h: SUB PIXEL COLORS
3278                                    <1>     ;              04h: OR PIXEL COLORS
3279                                    <1>     ;              05h: AND PIXEL COLORS
3280                                    <1>     ;              06h: XOR PIXEL COLORS
3281                                    <1>     ;              0Bh: MIX PIXEL COLORS
3282                                    <1>     ;                   CL = color (8 bit, 256 colors)
3283                                    <1>     ;                  ECX = color (16 bit and true colors)
3284                                    <1>     ;                  EDX = start position (row, column) (*)
3285                                    <1>     ;                       (HW = row, DX = column)
3286                                    <1>     ;                  ESI = size (rows, colums) (*)
3287                                    <1>     ;                        (HW = rows, SI = columns)
3288                                    <1>     ;                  EDI = mask color (**) ; 25/02/2021
3289                                    <1>     ;                        (this color will be excluded)
3290                                    <1>     ;              07h: NOT PIXEL COLORS
3291                                    <1>     ;              08h: NEG PIXEL COLORS
3292                                    <1>     ;              09h: INC PIXEL COLORS
3293                                    <1>     ;              0Ah: DEC PIXEL COLORS
3294                                    <1>     ;                  ECX = start position (row, column) (*)
3295                                    <1>     ;                        (HW = row, CX = column)
3296                                    <1>     ;                  EDX = size (rows, colums) (*)
3297                                    <1>     ;                        (HW = rows, DX = columns)
3298                                    <1>     ;                         (0 -> invalid)
3299                                    <1>     ;                         (1 -> horizontal or vertical line)
3300                                    <1>     ;                  EDI = mask color (**) ; 25/02/2021
3301                                    <1>     ;                        (this color will be excluded)
3302                                    <1>     ;              0Ch: REPLACE PIXEL COLORS
3303                                    <1>     ;                   CL = current color (8 bit, 256 colors)
3304                                    <1>     ;                  ECX = current color (16 bit and true colors)
3305                                    <1>     ;                   DL = new color (8 bit, 256 colors)
3306                                    <1>     ;                  EDX = new color (16 bit and true colors)
3307                                    <1>     ;                  ESI = start position (row, column) (*)
3308                                    <1>     ;                        (HW = row, SI = column)
3309                                    <1>     ;                  EDI = size (rows, colums) (*)
3310                                    <1>     ;                        (HW = rows, DI = columns)
3311                                    <1>     ;              0Dh: COPY PIXEL BLOCK(S) -full screen-
3312                                    <1>     ;                 -If BL bit 5 is 0-
3313                                    <1>     ;                  ECX = start position (row, column) (*)
```

```
                        ;                   (HW = row, CX = column)
                        ;               EDX = size (rows, colums) (*)
                        ;                   (HW = rows, DX = columns)
                        ;                    (0 -> invalid)
                        ;                    (1 -> horizontal or vertical line)
                        ;               ESI = destination (row, column) (***)
                        ;              -If BL bit 5 is 1-
                        ;                   CL = color (8 bit, 256 colors)
                        ;                 ECX = color (16 bit and true colors)
                        ;                 EDX = count of blocks (not bytes)
                        ;                       (limit: 2048 blocks)
                        ;                 ESI = user's buffer address
                        ;                     contains 64 bits block data
                        ;                     BLOCK ADDRESS - (row, col), dword
                        ;                     (first 32 bits)
                        ;                     BLOCK SIZE - (rows, cols), dword
                        ;                     (second 32 bits)
                        ;             ; 10/02/2021
                        ;             0Eh: WRITE LINE(s) -full screen-
                        ;                -If BL bit 5 is 0-
                        ;                 CL = color (8 bit, 256 colors)
                        ;                ECX = color (16 bit and true colors)
                        ;                 DX = low 12 bits - size (length)
                        ;                      high 4 bits - direction or type
                        ;                          0 - Horizontal line
                        ;                          1 - Vertical line
                        ;                       > 1 - undefined, invalid
                        ;                ESI = start position (row, column)
                        ;                      (HW = row, SI = column)
                        ;                -If BL bit 5 is 1-
                        ;                  CL = color (8 bit, 256 colors)
                        ;                 ECX = color (16 bit and true colors)
                        ;                  DX = number of lines (in user buffer)
                        ;                       (limit: 2048 lines)
                        ;                 ESI = user's buffer
                        ;                       contains 64 bit data for lines
                        ;                       START POINT: 32 bit (row, col)
                        ;                       LENGTH: 32 bit
                        ;                             high 16 bits - 0
                        ;                             bit 0-11 - length
                        ;                             bit 12-15 - type (length)
                        ;             0Fh: WRITE CHARACTER (FONT)
                        ;                  CL = char's color (8 bit, 256 colors)
                        ;                 ECX = char's color (16 bit and true colors)
                        ;                  DL = Character's ASCII code
                        ;                  DH bit 0 -> font height
                        ;                       0 -> 8x16 character font
                        ;                       1 -> 8x8 character font
                        ;                  DH bit 1 & 2 -> scale
                        ;                       0 = 1/1 (8 pixels per char row)
                        ;                       1 = 2/1 (16 pixels per char row)
                        ;                       2 = 3/1 (24 pixels per char row)
                        ;                       3 = 4/1 (32 pixels per char row)
                        ;                  DH bit 6 -> [ufont] option (1 = use [ufont])
                        ;                  If DH bit 7 = 1
                        ;                     USER FONT (from user buffer)
                        ;                        DL = 0 -> 8x8 (width: 1 byte per row)
                        ;                        DL = 1 -> 8x16
                        ;                        DL = 2 -> 16x16 (width: 2 bytes)
                        ;                        DL = 3 -> 16x32
                        ;                        DL = 4 -> 24x24 (width: 3 bytes)
                        ;                        DL = 5 -> 24x48
                        ;                        DL = 6 -> 32x32 (width: 4 bytes)
                        ;                        DL = 7 -> 32x64
                        ;                        DL > 7 -> invalid (unused)
                        ;                     EDI = user's font buffer address
                        ;                        (NOTE: byte order is as row0,row1,row2..)
                        ;                 ESI = start position (row, column) (*)
                        ;                       (HW = row, SI = column)
                        ;
                        ;             -- system to user --
                        ;                BL (bit 0 to 7)
                        ;             40h: COPY PIXELS (full screen, display page)
                        ;                 EDI = user's buffer address
                        ;             41h: COPY PIXELS (window)
                        ;                 ECX = start position (row, column) (*)
                        ;                       (HW = row, CX = column)
                        ;                 EDX = size (rows, colums) (*)
                        ;                       (HW = rows, DX = columns)
                        ;                        (<=1 -> horizontal or vertical line)
                        ;                 EDI = user's buffer address
                        ;
                        ;             Example: (29/01/2021)
                        ;                 ecx = 00400064h (start at row 64, column 100)
                        ;                 edx = 00320048h (size: 50 rows, 72 columns)
                        ;                                 (end at row 114, column 172)
                        ;                 If video memory starts at 0A0000h
                        ;                 and if resolution is 320x200 (256 colors) ..
                        ;                             window start offset: (64*320)+100 = 20580
                        ;                             window size: 16072 bytes (pixels)
                        ;                             window end offset: 20580+16072 = 36652
                        ;                             window start address: 0A0000h+564h = 0A5064h
                        ; Outputs:
                        ;       EAX = transfer/byte count
                        ;
                        ;       NOTE: If the source or destination address passes out of
                        ;       video pages (display memory limits), data will not be transferred
                        ;       and EAX will return as 0.
                        ;
                        ; 08/02/2021
                        ; 07/02/2021
                        ; 04/01/2021
                        ; PIXEL READ/WRITE (in current/active video mode)
                        ;
                        ;       BH = 3 = Read/Write pixel(s) -for all graphics modes-
                        ;            BL =
                        ;              0 = Read pixel
                        ;              1 = Write pixel
                        ;              2 = swap pixel colors
                        ;              3 = mix pixel colors
                        ;            29/01/2021
                        ;              4 = read pixels from user defined positions
                        ;              5 = write single color pixels to user defined positions
                        ;              6 = write multi color pixels to user defined positions
                        ;
                        ;             > 6 = invalid/unimplemented
                        ;
                        ;            .. for BL = 0 to 5
                        ;            CL = color for writing pixel(s) or
                        ;            ECX = color for writing pixel(s) in true color modes
                        ;
                        ;            EDX = Offset from start of video memory (0A0000h)
                        ;                  or start of linear frame buffer
                        ;
```

```
3438                            <1>     ;           07/02/2021
3439                            <1>     ;           .. for BL = 4
3440                            <1>     ;           EDI = user's destination buffer address for pixel colors
3441                            <1>     ;           29/01/2021
3442                            <1>     ;           .. for BL = 4 & 5
3443                            <1>     ;           ESI = user's source buffer address for BL = 4 & 5
3444                            <1>     ;           (buffer contains dword offset positions for pixels)
3445                            <1>     ;           EDX = number of pixels
3446                            <1>     ;           .. for BL = 6
3447                            <1>     ;           ESI = user's buffer address for BL = 6
3448                            <1>     ;           (buffer contains dword offset position and dword color
3449                            <1>     ;            value for each pixel)
3450                            <1>     ;           EDX = number of pixels
3451                            <1>     ;
3452                            <1>     ; Note:
3453                            <1>     ;     Pixel read/write will be performed in current video mode.
3454                            <1>     ;     If [CRT_MODE] < 0FFh, 0A0000h will be used
3455                            <1>     ;        as video memory and limit will be 65536
3456                            <1>     ;        (new/mix pixel color will be in CL)
3457                            <1>     ;     if [CRT_MODE] = 0FFh (VESA VBE video mode)
3458                            <1>     ;        LFB base address will be used as video memory
3459                            <1>     ;        and limit will be video page size
3460                            <1>     ;        (new/mix pixel color will be in CL)
3461                            <1>     ;
3462                            <1>     ; Outputs:
3463                            <1>     ;     EAX = pixel color (according to BL and ECX input)
3464                            <1>     ;     EAX = 0 (pixel color is 0 or there is an error)
3465                            <1>     ;          (BL will return as 0FFh if there is an error)
3466                            <1>     ;     ; 29/01/2021
3467                            <1>     ;     EAX = number of pixels (for BL input = 4&5&6)
3468                            <1>     ;
3469                            <1>     ; DIRECT (STANDARD VGA/CGA) DISPLAY MEMORY ACCESS FUNCTIONS:
3470                            <1>     ;
3471                            <1>     ;     BH = 4 = CGA direct video memory (0B8000h, 32K) access
3472                            <1>     ;            Page directory & page tables of the user's
3473                            <1>     ;            program will be updated to direct access to
3474                            <1>     ;            0B8000h (32K) video (CGA, color) memory; if
3475                            <1>     ;            there is not a permission conflict or lock!
3476                            <1>     ;             (User's program/process will have permission to
3477                            <1>     ;            access locked display memory if the owner is
3478                            <1>     ;            it's parent.)
3479                            <1>       ;
3480                            <1>     ;         Screen width = 320
3481                            <1>     ;
3482                            <1>     ;     BH = 5 = VGA direct video memory (0A0000h, 64K) access
3483                            <1>     ;            Page directory & page tables of the user's
3484                            <1>     ;            program will be updated to direct access to
3485                            <1>     ;            0A0000h (64K) video (VGA) memory; if there is not
3486                            <1>     ;            a permission conflict or lock!
3487                            <1>     ;             (User's program/process will have permission to
3488                            <1>     ;            access locked display memory if the owner is
3489                            <1>     ;            it's parent.)
3490                            <1>     ;
3491                            <1>     ;         ; 23/11/2020
3492                            <1>     ;         Screen width options = 320, 640, 800
3493                            <1>     ;
3494                            <1>     ; Outputs:
3495                            <1>     ;     EAX = Display memory address for direct access
3496                            <1>     ;           0A0000h for VGA, 0B8000h for CGA
3497                            <1>     ;     (Display memory size: 32K for CGA, 64K for VGA)
3498                            <1>     ;     EAX = 0 if display page access permission has been denied.
3499                            <1>     ;           (Locked!)
3500                            <1>     ;
3501                            <1>     ; LINEAR FRAME BUFFER ACCESS FUNCTIONS:
3502                            <1>     ;
3503                            <1>     ;     BH = 6 = Linear Frame Buffer direct video memory access
3504                            <1>     ;
3505                            <1>     ;            Page directory & page tables of the user's
3506                            <1>     ;            program will be updated to direct access to
3507                            <1>     ;            the configured LFB (Linear Frame Buffer) address,
3508                            <1>     ;            if there is not a permission conflict or lock!
3509                            <1>     ;             (User's program/process will have permission to
3510                            <1>     ;            access locked display memory if the owner is
3511                            <1>     ;            it's parent.)
3512                            <1>     ;
3513                            <1>     ;        ; 10/12/2020
3514                            <1>     ;        BL = 0FFh -> Direct LFB access for current video mode
3515                            <1>     ;        BL = XXh < 0FFh -> Direct LFB access
3516                            <1>     ;                           for VESA video mode 1XXh
3517                            <1>     ;
3518                            <1>     ;            Return: EAX = Linear Frame Buffer address
3519                            <1>     ;                    (EAX = 0 -> error)
3520                            <1>     ;                 If EAX > 0
3521                            <1>     ;                    EDX = Frame Buffer Size in bytes
3522                            <1>     ;                      BH = Requested Video Mode - 100h
3523                            <1>     ;                           (VESA VBE video modes)
3524                            <1>     ;                      BL = bits per pixel
3525                            <1>     ;                           8 = 256 colors, 8
3526                            <1>     ;                          16 = 65536 colors, 5-6(G)-5
3527                            <1>     ;                          24 = RGB, 16M colors, 8-8-8
3528                            <1>     ;                          32 = RGB + alpha bytes, 8-8-8-8
3529                            <1>     ;                 If BH = 0FFh
3530                            <1>     ;                    BL = VGA/CGA video mode (also EAX = 0)
3531                            <1>     ;
3532                            <1>     ;            ** Function will return with EAX = 0 if the mode
3533                            <1>     ;            is not a valid VESA VBE video mode as 1??h **
3534                            <1>     ;
3535                            <1>     ;            ECX = Pixel resolution
3536                            <1>     ;                  CX = Width (320, 640, 800, 1024, 1366, 1920)
3537                            <1>     ;                  High 16 bits of ECX = Height
3538                            <1>     ;
3539                            <1>     ;     23/11/2020
3540                            <1>     ; ***  GET VIDEO MODE & LINEAR FRAME BUFFER INFO
3541                            <1>     ;     (This function -7- also is used for VGA and CGA modes)
3542                            <1>     ;
3543                            <1>     ;     BH = 7 = Get Linear Frame Buffer info
3544                            <1>     ;
3545                            <1>     ;        ; 22/01/2021
3546                            <1>     ;        ; 10/12/2020
3547                            <1>     ;        BL = any -not used- (22/01/2021)
3548                            <1>     ;
3549                            <1>     ;            Return:
3550                            <1>     ;            EAX = Frame Buffer Address (0 = is not in use)
3551                            <1>     ;            EDX = Frame Buffer Size in bytes
3552                            <1>     ;             BH = Current Video Mode - 100h  ; 22/01/2021
3553                            <1>     ;                  (VESA VBE video modes)
3554                            <1>     ;             BL = bits per pixel
3555                            <1>     ;                   8 = 256 colors, 8
3556                            <1>     ;                  16 = 65536 colors, 5-6(G)-5
3557                            <1>     ;                  24 = RGB, 16M colors, 8-8-8
3558                            <1>     ;                  32 = RGB + alpha bytes, 8-8-8-8
3559                            <1>     ;            If BH = 0FFh
3560                            <1>     ;               BL = VGA/CGA video mode (also EAX = 0)
3561                            <1>     ;
```

```
3562                                <1>     ;             Note:
3563                                <1>     ;             Alpha byte will be used as virtual color index.
3564                                <1>     ;             (32 bit pixel colors.. byte 0,1,2 rgb and 3 alpha)
3565                                <1>     ;
3566                                <1>     ;             ** Function will return with EAX = 0 if the mode
3567                                <1>     ;             is not a valid VESA VBE video mode as 1??h **
3568                                <1>     ;
3569                                <1>     ;             ECX = Pixel resolution
3570                                <1>     ;                   CX = Width (320, 640, 800, 1024, 1366, 1920)
3571                                <1>     ;                   High 16 bits of ECX = Height
3572                                <1>     ;
3573                                <1>     ;       NOTE: Each process will have it's own frame buffer
3574                                <1>     ;             address and resolution parameters in 'u' area.
3575                                <1>     ;             Then, if the current frame buffer & resolution
3576                                <1>     ;             is different, frame buffer r/w functions
3577                                <1>     ;             will use scale factor to convert process's
3578                                <1>     ;                 pixel coordinates to actual screen coordinates.
3579                                <1>     ;             resolution -> dimensional scale
3580                                <1>     ;             color size -> color scale
3581                                <1>     ;            * RGB (TRUE) colors to 256 colors conversion:
3582                                <1>     ;                 TRUE Colors -> 8,8,8 (R,G,B; byte 0 is R)
3583                                <1>     ;             256 colors -> 2,2,2,2 (R,G,B,L; bit 0&1 is R)
3584                                <1>     ;                bit 6&7 -> luminosity base level (0,1,2,3)
3585                                <1>     ;                bit 4&5 -> blue level (0,1,2,3)
3586                                <1>     ;                bit 2%3 -> green level (0,1,2,3)
3587                                <1>     ;                bit 0&1 -> red level (0,1,2,3)
3588                                <1>     ;             Example: total red level : luminosity + red level
3589                                <1>     ;             Luminosity base level: 0 -> 16
3590                                <1>     ;                                    1 -> 32
3591                                <1>     ;                                    2 -> 64
3592                                <1>     ;                                    3 -> 128
3593                                <1>     ;             Color level:
3594                                <1>     ;                                   0 -> 0
3595                                <1>     ;                                   1 -> luminosity level
3596                                <1>     ;                                   2 -> luminosity level + 64
3597                                <1>     ;                                   3 -> 255
3598                                <1>     ;               Luminosity base level = min (R,G,B)
3599                                <1>     ;                        if it is <16, it will be set to 16
3600                                <1>     ;                Color levels: Color values are fixed to (nearest)
3601                                <1>     ;                     one of all possible set level (step) values
3602                                <1>     ;                     (according to luminosity base level); then
3603                                <1>     ;                     color levels are set to R-L, G-L, B-L.
3604                                <1>     ;              For example: If luminosity base level is 32
3605                                <1>     ;                       all possible set values are 0, 32, 96, 255.
3606                                <1>     ;
3607                                <1>     ;                * RGB (TRUE) colors to 16 colors conversion:
3608                                <1>     ;                16 colors: R,B,G,L bits (4 bits)
3609                                <1>     ;                      If any one of R,G,B >= 128 L = 1
3610                                <1>     ;                      If max. value of (R,G,B) >= 32, it is 1
3611                                <1>     ;                         else all color bits (R&G&B&L) are 0
3612                                <1>     ;                      If the second value >= max. value / 2
3613                                <1>     ;                         it is 1
3614                                <1>     ;                      If the third value >= max. value / 2
3615                                <1>     ;                         it is 1
3616                                <1>     ;                Example: R = 132, G = 64, B = 78
3617                                <1>     ;                        L = 1, R = 1
3618                                <1>     ;                        G < 66 --> G = 0
3619                                <1>     ;                        B >= 66 --> B = 1
3620                                <1>     ;
3621                                <1>     ; 10/12/2020
3622                                <1>     ; SET VIDEO MODE (& RETURN LFB INFO for VESA VBE VIDEO MODES)
3623                                <1>     ;
3624                                <1>     ;       BH = 8 = Set Video Mode
3625                                <1>     ;
3626                                <1>     ;              BL = Requested Video Mode (method)
3627                                <1>     ;              If BL = 0FFh
3628                                <1>     ;                 CX = VESA VBE Video Mode
3629                                <1>     ;                 ; 11/12/2020
3630                                <1>     ;                 If EDX > 0 -> LFB INFO (user) buffer addr
3631                                <1>     ;              If BL < 0FFh, it is VGA/CGA video mode and
3632                                <1>     ;                    CX & EDX will not be used
3633                                <1>     ;
3634                                <1>     ;             NOTE: The last VESA VBE video mode is 11Bh but
3635                                <1>     ;             TRDOS 386 will permit to set video mode upto 11Fh.
3636                                <1>     ;             Above 11Fh, from 140h to 1FEh, it will be accepted
3637                                <1>     ;             as Bochs/Plex86 emulator video mode and it will be
3638                                <1>     ;             used only if [vbe3] = 2 and detected
3639                                <1>     ;             video bios is BOCHS/PLEX86/QEMU/VIRTUALBOX vbios.
3640                                <1>     ;
3641                                <1>     ; Outputs:
3642                                <1>     ;       EAX = Requested (Proper) video mode number + 1
3643                                <1>     ;             ("dec eax" by user will give requested video mode),
3644                                <1>     ;
3645                                <1>     ;       If BL input is 0FFh
3646                                <1>     ;          EDX = LFBINFO table/structure (in user's buffer addr)
3647                                <1>     ;          EDX = 0 -> Invalid LFBINFO (do not use it)
3648                                <1>     ;
3649                                <1>     ;       EAX = 0 -> Error (but EDX will not be changed)
3650                                <1>     ;
3651                                <1>     ; 03/12/2020
3652                                <1>     ; VESA VBE3 VIDEO BIOS (32 bit) PROTECTED MODE INTERFACE SETTINGS
3653                                <1>     ;
3654                                <1>     ;       BH = 9 = set/get VBE3 Protected Mode Interface parameters
3655                                <1>     ;
3656                                <1>     ;              BL = 0 - Disable protected mode interface
3657                                <1>     ;                     ([pmi32] = 0)
3658                                <1>     ;                     Return: AL = 1
3659                                <1>     ;              BL = 1 - Enable protected mode Interface
3660                                <1>     ;                     ([pmi32] = 1)
3661                                <1>     ;                     Return: AL = 2
3662                                <1>     ;              BL = 2 - Get protected mode interface status
3663                                <1>     ;                     Return: AL = [pmi32] + 1 (AL = 1 or 2)
3664                                <1>     ;
3665                                <1>     ;              If [vbe3] <> 3 --> AL = 0
3666                                <1>     ;
3667                                <1>     ;              ; 17/01/2021
3668                                <1>     ;              BL = 3 - Disable/Cancel restore permission to user
3669                                <1>     ;                     Return: AL = 1 (if disabled) or 0
3670                                <1>     ;              BL = 4 - Enable/Give restore permission to user
3671                                <1>     ;                     Return: AL = 2 (if enabled) or 0
3672                                <1>     ;              BL = 5 - Get video state save/restore status
3673                                <1>     ;                      (permission status)
3674                                <1>     ;                      Return: AL = Status (enabled = 1)
3675                                <1>     ;                              ; 22/01/2021
3676                                <1>     ;                             AH = state options ([srvso])
3677                                <1>     ;              BL = 6 - Return VESA VBE number/status
3678                                <1>     ;                      Return: AX = status
3679                                <1>     ;                           if AH =  2, AL > 0 : Emulator
3680                                <1>     ;                             AH =  3, VESA VBE3 video bios
3681                                <1>     ;              ; 28/02/2021
3682                                <1>     ;              BL = 7 - Set true color mode as 32bpp (default)
3683                                <1>     ;                     Return: AX = 32 (if VBE3)
3684                                <1>     ;              NOTE: Initial/default value is 32bpp for vbe3.
3685                                <1>     ;                     Return: AX = 0 -> error
```

```
                        ;               BL = 8 - Set true color mode as 24bpp (default)
                        ;                       Return: AX = 24
                        ;                       ;Return: AX = 0 -> error
                        ;               BL = 9 - Return default/current true color mode
                        ;                       Return: AX = 32 (32 bpp)
                        ;                       Return: AX = 24 (24 bpp)
                        ;                       Return: AX = 0 -> error (not VESA bios)
                        ;
                        ;               BL > 9 : not implemented (28/02/2021)
                        ;
                        ;       ; 19/01/2021 ([u.uid] check)
                        ;       Note: Enabling/Disabling are done by root ([u.uid] = 0)
                        ;             while [multi_tasking] = 0.
                        ;
                        ; 23/12/2020
                        ; VIDEO MEMORY MAPPING:
                        ;       BH = 10 = Map video memory to user's buffer
                        ;
                        ;          BL = 0 : CGA memory (0B8000h) map (32K)
                        ;          BL = 1 : VGA memory (0A0000h) map (64K)
                        ;          BL = 2 : SVGA memory (LFB) map to user's buffer
                        ;
                        ;       ECX = User's buffer addr (low 12 bits will be cleared)
                        ;       EDX = Buffer size in bytes (if BL = 2)
                        ;              (will be trimmed if LFB size < EDX)
                        ; Return:
                        ;   EAX = physical address of video memory (buffer)
                        ;       EBX = mapped (actual) size of video memory (bytes)
                        ;       ECX = virtual start address of user's video buffer
                        ;       EDX is same with EDX input
                        ;
                        ;       (Note: Memory page boundaries will be applied
                        ;        to buffer size and buff start addr by rounding down.
                        ;        Rounded size & address values must not be zero.)
                        ;       -Normally, it is expected to request mapping by using
                        ;        correct buffer size of current or desired video mode-
                        ;
                        ;       EAX = 0 -> error ! memory can not mapped to user
                        ;
                        ; 04/01/2021
                        ; SET/READ COLOR PALETTE (set/read DAC color registers)
                        ;       ((256 colors (8bpp) VGA/CGA video hardware feature))
                        ;
                        ;       BH = 11 = Set/Read DAC color registers
                        ;
                        ;          (BL<4 Original method for std VGA video hardware)
                        ;          BL = 0 : Read all DAC color registers (256 colors)
                        ;              (6 bit colors, in RGB order)
                        ;          BL = 1 : Set all DAC color registers (256 colors)
                        ;              (6 bit colors, in RGB order)
                        ;          BL = 2 : Read single DAC color register
                        ;              (6 bit color, in RGB order)
                        ;              ((EAX will return with color value))
                        ;              CL = DAC color register (index)
                        ;          BL = 3 : Set/Write single DAC color register
                        ;              (6 bit color, in RGB order, bit 6&7 are 0)
                        ;              ECX byte 0 - DAC color register
                        ;              ECX byte 1 - Red (6 bit)
                        ;              ECX byte 2 - Green (6 bit)
                        ;              ECX byte 3 - Blue (6 bit)
                        ;          (BL>3 Alternative method for BMP files etc.)
                        ;          BL = 4 : Read all DAC color registers (256 colors)
                        ;              (8 bit colors, in BGR order, bit 0&1 is 0)
                        ;          BL = 5 : Set all DAC color registers (256 colors)
                        ;              (8 bit colors, in BGR order, bit 0&1 is 0)
                        ;          BL = 6 : Read single DAC color register
                        ;              (8 bit color, in BGR order, bit 0&1 is 0)
                        ;              ((EAX will return with color value))
                        ;              CL = DAC color register (index)
                        ;          BL = 7 : Set/Write single DAC color register
                        ;              (8 bit color, bit 0&1 are 0)
                        ;              ECX byte 0 - DAC color register
                        ;              ECX byte 1 - Blue (8 bit)
                        ;              ECX byte 2 - Green (8 bit)
                        ;              ECX byte 3 - Red (8 bit)
                        ;
                        ;            BL > 7 : invalid (not implemented)
                        ;
                        ;     if BL bit 2 is 1, 6 bit colors converted to 8 bit colors
                        ;        (low two bits of color bytes will be 0)
                        ;       ((color byte 00111111b will be converted to 11111100b))
                        ;        and RGB byte order will be
                        ;                byte 0 - Blue (low 2 bits are 0)
                        ;                byte 1 - Green (low 2 bits are 0)
                        ;                byte 2 - Red (low 2 bits are 0)
                        ;                byte 3 - pad (or zero byte)
                        ;        and 256 colors buffer size must be 256*4 = 1024 bytes
                        ;     if BL bit 2 is 0, 6 bit colors will be used directly
                        ;           (high two bits of 8 bit color bytes will be 0)
                        ;        ((dac color 111111b will be converted to 00111111b))
                        ;                byte 0 - Red (high 2 bits are 0)
                        ;                byte 1 - Green (high 2 bits are 0)
                        ;                byte 2 - Blue (high 2 bits are 0)
                        ;         and 256 colors buffer size must be 256*3 = 768 bytes
                        ;
                        ;       ECX = User's buffer addr (256*3 = 768 bytes) or
                        ;             Color
                        ;
                        ; Return:
                        ;     ;   EAX = buffer size (for BL input = 0,1,4,5)
                        ;               or color value (for BL input = 2,3,6,7)
                        ;
                        ; 10/01/2021
                        ; SET/READ FONT DATA
                        ;
                        ;       BH = 12 = Set/Read Character Font Data
                        ;
                        ;          BL = 0 : Disable system font overwrite
                        ;          BL = 1 : Enable system font overwrite
                        ;          BL = 2 : Read system font 8x8
                        ;          BL = 3 : Read system font 8x14
                        ;          BL = 4 : Read system font 8x16
                        ;          BL = 5 : Read user defined font 8x8
                        ;          BL = 6 : Read user defined font 8x16
                        ;          BL = 7 : Write system font 8x8
                        ;          BL = 8 : Write system font 8x14
                        ;          BL = 9 : Write system font 8x16
                        ;          BL = 10 : Write user defined font 8x8
                        ;          BL = 11 : Write user defined font 8x16
                        ;
                        ;        BL > 11 : invalid (not implemented)
                        ;
                        ;       For BL = 1 to 11
                        ;          ECX = number of characters (<= 256)
```

```
                        <1>     ;         EDX = first character (ascii code in DL)
                        <1>     ;         ESI = user's buffer address
                        <1>     ;
                        <1>     ; Return:
                        <1>     ;   EAX = number of characters (ecx input)
                        <1>     ;       EAX = 0 -> error
                        <1>     ;       (EAX = 256 for BL = 0 and 1 if successful)
                        <1>     ;
                        <1>     ; Note: system font overwrite permission will be
                        <1>     ;        given if [multi_tasking] = 0
                        <1>     ;       and [u.uid] = 0 (BL = 1) ; 19/01/2021
                        <1>     ;        and if [ufont] bit 7 is 1 (BL = 7,8,9)
                        <1>     ;
                        <1>     ; 18/01/2021
                        <1>     ; SAVE/RESTORE STANDARD VGA VIDEO STATE
                        <1>     ;
                        <1>     ;       BH = 13 = Save/Restore std VGA video state
                        <1>     ;
                        <1>     ;          BL = 0 : Save VGA state (without DAC regs)
                        <1>     ;               Return: EAX = VideoStateID (>0)
                        <1>     ;                       EAX = 0 (failed!)
                        <1>     ;              (size: 110 bytes for TRDOS 386 v2.0.3)
                        <1>     ;          BL = 1 : Restore VGA state (without DAC regs)
                        <1>     ;               ECX = VideoStateID (to be verified)
                        <1>     ;                Return: EAX = Restore size (>0)
                        <1>     ;          BL = 2 : Save VGA state (complete)
                        <1>     ;               Return: EAX = VideoStateID (>0)
                        <1>     ;                       EAX = 0 (failed!)
                        <1>     ;              (size: 882 bytes for TRDOS 386 v2.0.3)
                        <1>     ;          BL = 3 : Restore VGA state (complete)
                        <1>     ;               ECX = VideoStateID (to be verified)
                        <1>     ;                Return: EAX = Restore size (>0)
                        <1>     ;
                        <1>     ;       * Above options are for saving
                        <1>     ;       *       video state to system memory
                        <1>     ;       *       (location: VBE3VIDEOSTATE, 2048 bytes)
                        <1>     ;
                        <1>     ;          BL = 4 : Save VGA state (without DAC regs)
                        <1>     ;               ECX = buffer address
                        <1>     ;               Return: EAX = transfer count
                        <1>     ;              (size: 110 bytes for TRDOS 386 v2.0.3)
                        <1>     ;               ECX = buffer address
                        <1>     ;          BL = 5 : Restore VGA state (without DAC regs)
                        <1>     ;               ECX = buffer address
                        <1>     ;                Return: EAX = transfer count
                        <1>     ;          BL = 6 : Save VGA state (complete)
                        <1>     ;               ECX = buffer address
                        <1>     ;               Return: EAX = transfer count
                        <1>     ;              (size: 882 bytes for TRDOS 386 v2.0.3)
                        <1>     ;          BL = 7 : Restore VGA state (complete)
                        <1>     ;               ECX = buffer address
                        <1>     ;                Return: EAX = transfer count
                        <1>     ;
                        <1>     ;       * Above options are for saving
                        <1>     ;       *       video state to user's buffer
                        <1>     ;       *       (buffer size: 110 bytes or 882 bytes)
                        <1>     ;
                        <1>     ;         BL > 7 : invalid (not implemented)
                        <1>     ;
                        <1>     ; 18/01/2021
                        <1>     ; SAVE/RESTORE SUPER VGA (VESA VBE 2/3) VIDEO STATE
                        <1>     ;
                        <1>     ;       BH = 14 = Save/Restore SVGA video state
                        <1>     ;
                        <1>     ;          BL = options
                        <1>     ;               bit 0 - Save (0) or Restore (1)
                        <1>     ;               bit 1 - controller hardware state
                        <1>     ;               bit 2 - BIOS data state
                        <1>     ;               bit 3 - DAC state
                        <1>     ;               bit 4 - (extended) Register state
                        <1>     ;               bit 5 - system (0) or user (1) memory
                        <1>     ;               bit 6 - verify without transfer
                        <1>     ;               bit 7 - not used (must be 0)
                        <1>     ;
                        <1>     ;            if bit 0 = 0 and bit 5 = 0
                        <1>     ;              Return: EAX = VideoStateID (>0)
                        <1>     ;            if bit 0 = 1
                        <1>     ;              ECX = VideoStateID (bit 5 = 0)
                        <1>     ;              Return: EAX = restore (transfer) size
                        <1>     ;            if bit 5 = 1
                        <1>     ;              ECX = Buffer address
                        <1>     ;              Return: EAX = transfer count (size)
                        <1>     ;
                        <1>     ;            ECX = Buffer address or VideoStateID
                        <1>     ;
                        <1>     ;         BL > 127 : invalid (not implemented)
                        <1>     ;
                        <1>     ;  Note: Required buffer size may be > 2048 bytes
                        <1>     ;        (function fails when buff size > 2048 bytes)
                        <1>     ;        proper option must be used
                        <1>     ;
                        <1>     ; 18/01/2021
                        <1>     ; READ VESA EDID (EXTENDED DISPLAY IDENTIFICATION DATA)
                        <1>     ;
                        <1>     ;       BH = 15 = Read VESA EDID for connected monitor
                        <1>     ;                 (copy EDID to user)
                        <1>     ;
                        <1>     ;          BL = any
                        <1>     ;
                        <1>     ;       Input:
                        <1>     ;           ECX = user's (EDID) buffer address
                        <1>     ;                 (buffer size: 128 bytes)
                        <1>     ;       Output:
                        <1>     ;           EAX = 128 (EDID size)
                        <1>     ;           or EAX = 0 -> Error!
                        <1>     ;              (EDID not ready or buffer addr error)
                        <1>     ;
                        <1>
                        <1>     ; 16/05/2016
0000D3CC 31C0           <1>     xor     eax, eax
0000D3CE A3[348E0100]   <1>     mov     [u.r0], eax
0000D3D3 20FF           <1>     and     bh, bh
                        <1>     ;jnz    sysvideo_13 ; 11/07/2016
                        <1>     ; 23/07/2022
0000D3D5 7405           <1>     jz      short sysvideo_40
0000D3D7 E974020000     <1>     jmp     sysvideo_13
                        <1>
                        <1> sysvideo_40:        ; 23/07/2022
                        <1>     ;; 21/11/2020 (TRDOS 386 v2.0.3)
                        <1>     ;; tty/text mode transfers are only for video mode 3
                        <1>
                        <1>     ; 22/11/2020
                        <1>     ;cmp    byte [CRT_MODE], 3 ; 80x25 text, 16 colors
                        <1>     ;jne    sysret ; invalid (nothing to do), [u.r0] = 0
```

```
                                <1>
                                <1>     ; 23/11/2020
                                <1>     ; bit 7,6,5,4 of BL are reserved and it must be 0
                                <1>     ;               for current 'sysvideo' version
                                <1>     ;test  bl, 0F0h
                                <1>     ;;jnz  sysret ; invalid (undefined) !
                                <1>     ;; 28/01/2021
                                <1>     ;jnz   short sysvideo_1_2 ; invalid (undefined) !
                                <1>     ; 28/01/2021
0000D3DC 80FB07                 <1>     cmp    bl, 7
0000D3DF 776E                   <1>     ja     short sysvideo_1_2 ; invalid (undefined) !
                                <1>
                                <1>     ; Video mode 0, 80*25 text mode, CGA 16 colors
                                <1>     ; [CRT_MODE] = 3
                                <1>     ;mov   bh, bl
                                <1>     ;shr   bh, 2 ; 4..7 -> 1, 8..11 -> 2, 12..15 -> 3
                                <1>     ;; 21/11/2020
                                <1>     ;;and  bh, bh
                                <1>         ;jnz       sysvideo_4  ; Display page window transfer etc.
                                <1>
                                <1>     ; 28/01/2021
0000D3E1 F6C304                 <1>     test   bl, 4 ; bit 2
                                <1>     ;jnz   sysvideo_4 ; Display page window transfer
                                <1>     ; 23/07/2022
0000D3E4 7405                   <1>     jz     short sysvideo_41
0000D3E6 E9A1000000             <1>     jmp    sysvideo_4
                                <1> sysvideo_41:       ; 23/07/2022
                                <1>     ; Display page (complete) transfer
0000D3EB 80FA07                 <1>     cmp    dl, 7
                                <1>     ;jnz   sysret ; invalid (nothing to do), [u.r0] = 0
0000D3EE 760A                   <1>     jna    short sysvideo_0 ; 28/01/2021
0000D3F0 FEC2                   <1>     inc    dl ; 0FFh -> 0 ("use current video page")
0000D3F2 755B                   <1>     jnz    short sysvideo_1_2  ; invalid
                                <1>     ; dl = 0 -> use current current page
0000D3F4 8A15[AE760100]         <1>     mov    dl, [ACTIVE_PAGE]
                                <1> sysvideo_0:
                                <1>     ; 28/01/2021
0000D3FA 80FB03                 <1>     cmp    bl, 3
0000D3FD 7206                   <1>     jb     short sysvideo_0_0
0000D3FF 09FF                   <1>     or     edi, edi
0000D401 744C                   <1>     jz     short sysvideo_1_2  ; invalid
0000D403 89FE                   <1>     mov    esi, edi ; save swap/exchange buffer addr
                                <1>     ; ecx = user buffer for new video page content
                                <1>     ; esi = user (swap) buffer for saving current video page
                                <1> sysvideo_0_0:
0000D405 BF00800B00             <1>     mov    edi, 0B8000h
                                <1>     ; dl = display page number, destination
0000D40A 66B80010               <1>     mov    ax, 4096 ; 21/11/2020
0000D40E 20D2                   <1>     and    dl, dl
0000D410 7408                   <1>     jz     short sysvideo_1
                                <1>     ; 07/02/2021
0000D412 88D6                   <1>     mov    dh, dl
                                <1> sysvideo_0_1:
                                <1>     ; page length = 4096 bytes (but page content is 80*25*2 bytes)
0000D414 01C7                   <1>     add    edi, eax ; 21//11/2020 ([CRT_LEN] = 1000h for mode 3)
0000D416 FECE                   <1>     dec    dh
0000D418 75FA                   <1>     jnz    short sysvideo_0_1
                                <1> sysvideo_1:
0000D41A 80E303                 <1>     and    bl, 3
0000D41D 7535                   <1>     jnz    short sysvideo_2 ; user to system display page transfer
                                <1>     ; system to system video page (content) transfer
                                <1>     ; cl = display page number, source
0000D41F 80F907                 <1>     cmp    cl, 7
                                <1>     ;ja    sysret ; invalid (nothing to do), [u.r0] = 0
0000D422 760A                   <1>     jna    short sysvideo_1_0
0000D424 FEC1                   <1>     inc    cl ; 0FFh -> 0 ("use current video page")
0000D426 7527                   <1>     jnz    short sysvideo_1_2  ; invalid
0000D428 8A0D[AE760100]         <1>     mov    cl, [ACTIVE_PAGE]
                                <1> sysvideo_1_0:
                                <1>     ; 28/01/2021
0000D42E 38D1                   <1>     cmp    cl, dl
0000D430 741D                   <1>     je     short sysvideo_1_2  ; same video page !
                                <1>
                                <1>     ; system to system video/display page transfer (mode 0)
                                <1>     ; 21/11/2020
                                <1>     ;mov   esi, 0B8000h
                                <1>     ;movzx eax, cl
                                <1>     ;mov   edx, 4096 ; [CRT_LEN] = 1000h for video mode 3
                                <1>     ;mov   ecx, edx
                                <1>     ;mul   edx
                                <1>     ;add   esi, eax
                                <1>     ; 28/01/2021
                                <1>     ;movzx esi, cl
                                <1>     ;shl   si, 12 ; * 4096
                                <1>     ;add   esi, 0B8000h
                                <1>
                                <1>     ; 28/01/2021
0000D432 A3[348E0100]           <1>     mov    [u.r0], eax ; 4096
0000D437 BE00800B00             <1>     mov    esi, 0B8000h
0000D43C 08C9                   <1>     or     cl, cl
0000D43E 7409                   <1>     jz     short sysvideo_1_1
                                <1>     ; 07/02/2021
                                <1>     ;mov   al, cl ; display/video page
                                <1>     ;xor   ah, ah
                                <1>     ;shl   ax, 12 ; * 4096
                                <1>     ; 23/07/2022
0000D440 30C0                   <1>     xor    al, al
                                <1>     ;xor   eax, eax
0000D442 88CC                   <1>     mov    ah, cl ; CL * 256
0000D444 C1E004                 <1>     shl    eax, 4 ; * 16 = CL * 4096
0000D447 01C6                   <1>     add    esi, eax
                                <1> sysvideo_1_1:
                                <1>     ; 21/11/2020
                                <1>     ;;mov  ecx, 4096
                                <1>     ;mov   ecx, eax ; 4096
                                <1>     ;;mov  [u.r0], ecx ; 4096 bytes
                                <1>     ; 28/01/2021
                                <1>     ;mov   [u.r0], cx
0000D449 31C9                   <1>     xor    ecx, ecx
0000D44B B504                   <1>     mov    ch, 4 ; mov ecx, 1024
                                <1>     ;shr   cx, 2 ; / 4
0000D44D F3A5                   <1>     rep    movsd
                                <1> sysvideo_1_2:
0000D44F E92AF6FFFF             <1>     jmp    sysret
                                <1> sysvideo_2:
0000D454 80FB02                 <1>     cmp    bl, 2
                                <1>         ;ja sysret; invalid (user to user), [u.r0] = 0
                                <1>     ; 28/01/2021
0000D457 7226                   <1>     jb     short sysvideo_3 ; user to system
0000D459 7404                   <1>     je     short sysvideo_2_0 ; system to user
                                <1>     ; bl = 3
0000D45B 89CA                   <1>     mov    edx, ecx ; save user's buffer addr
0000D45D 89F1                   <1>     mov    ecx, esi ; save swap address
                                <1> sysvideo_2_0:
```

```
4058                                <1>     ; bl = 2 (or bl = 3, stage 1)
4059                                <1>     ; system to user video/display page transfer (mode 0)
4060 0000D45F 89FE                 <1>     mov     esi, edi
4061 0000D461 89CF                 <1>     mov     edi, ecx ; user buffer ; 28/01/2021
4062                                <1>     ; 21/11/2020
4063 0000D463 89C1                 <1>     mov     ecx, eax ; 4096
4064 0000D465 E880360000           <1>     call    transfer_to_user_buffer ; fast transfer
4065                                <1>     ;jc     sysret ; [u.r0] = 0
4066 0000D46A 72E3                 <1>     jc      short sysvideo_1_2 ; 28/01/2021
4067                                <1>     ; 28/01/2021
4068 0000D46C 80FB03               <1>     cmp     bl, 3
4069 0000D46F 7408                 <1>     je      short sysvideo_2_2
4070                                <1> sysvideo_2_1:
4071                                <1>     ; 21/11/2020
4072 0000D471 890D[348E0100]       <1>     mov     [u.r0], ecx
4073                                <1>     ;;mov   [u.r0], cx
4074                                <1>     ;jmp    sysret
4075 0000D477 EBD6                 <1>     jmp     short sysvideo_1_2
4076                                <1>
4077                                <1> sysvideo_2_2:
4078                                <1>     ; bl = 3 (exchange/swap) complete display page
4079                                <1>     ; esi = video page start address
4080                                <1>     ; edx = user's buffer address
4081                                <1>
4082                                <1>     ;mov    ecx, 4096
4083 0000D479 89F7                 <1>     mov     edi, esi ; video page start address
4084 0000D47B 89D6                 <1>     mov     esi, edx ; user's (new page) buffer address
4085 0000D47D EB04                 <1>     jmp     short sysvideo_2_3
4086                                <1> sysvideo_3:
4087                                <1>     ; bl = 1 (or bl = 3, stage 2)
4088                                <1>     ; user to system video/display page transfer (mode 0)
4089 0000D47F 89CE                 <1>     mov     esi, ecx ; user buffer
4090                                <1>     ; edi = video page address
4091                                <1>     ; 21/11/2020
4092 0000D481 89C1                 <1>     mov     ecx, eax ; 4096
4093                                <1> sysvideo_2_3:
4094 0000D483 E8AC360000           <1>     call    transfer_from_user_buffer ; fast transfer
4095                                <1>     ;jc     sysret ; [u.r0] = 0
4096 0000D488 72C5                 <1>     jc      short sysvideo_1_2 ; 28/01/2021
4097 0000D48A EBE5                 <1>     jmp     short sysvideo_2_1
4098                                <1>
4099                                <1>     ; 21/11/2020
4100                                <1>     ;mov    [u.r0], ecx
4101                                <1>     ;;mov   [u.r0], cx
4102                                <1>     ;;jmp   sysret
4103                                <1>     ;jmp    short sysvideo_1_2 ; 28/01/2021
4104                                <1>
4105                                <1> sysvideo_4:
4106                                <1>     ; 23/07/2022
4107                                <1>     ; 23/11/2020 (TRDOS 386 v2.0.3)
4108                                <1>
4109                                <1>     ; Display page window transfer etc.
4110 0000D48C 80E303               <1>     and     bl, 3
4111                                <1>         ;jnz       sysvideo_9 ; user to system or system to user
4112                                <1>     ; 23/07/2022
4113 0000D48F 7405                 <1>     jz      short sysvideo_4_0
4114 0000D491 E9E3000000           <1>     jmp     sysvideo_9
4115                                <1> sysvideo_4_0:
4116                                <1>     ; 21/11/2020
4117                                <1>     ; system to system video/display page window transfer (mode 0)
4118 0000D496 81FE00800B00         <1>     cmp     esi, 0B8000h   ; source buffer address (system)
4119                                <1>     ;jb     sysret
4120                                <1>     ; 23/07/2022
4121 0000D49C 7245                 <1>     jb      short sysvideo_4_3  ; jmp sysret
4122 0000D49E 81FE00000C00         <1>     cmp     esi, 0B8000h+(4096*8)
4123                                <1>     ; 23/07/2022
4124                                <1>     ;jnb    sysret
4125 0000D4A4 733D                 <1>     jnb     short sysvideo_4_3  ; jmp sysret
4126 0000D4A6 81FF00800B00         <1>     cmp     edi, 0B8000h   ; destination buffer address (system)
4127                                <1>     ;jb     sysret
4128                                <1>     ; 23/07/2022
4129 0000D4AC 7235                 <1>     jb      short sysvideo_4_3  ; jmp sysret
4130 0000D4AE 81FF00000C00         <1>         cmp     edi, 0B8000h+(4096*8)
4131                                <1>     ; 23/07/2022
4132                                <1>     ;jnb    sysret
4133 0000D4B4 732D                 <1>     jnb     short sysvideo_4_3  ; jmp sysret
4134                                <1>     ;
4135 0000D4B6 51                   <1>     push    ecx ; X1 and Y1 position - top left column, row
4136 0000D4B7 0FB7C1               <1>     movzx   eax, cx ; top left column
4137                                <1>     ; 21/11/2020
4138 0000D4BA C1E910               <1>     shr     ecx, 16 ; top row
4139 0000D4BD 740E                 <1>     jz      short sysvideo_4_1 ; bypass following code
4140 0000D4BF 52                   <1>     push    edx ; X2 and Y2 position - bottom right column, row
4141 0000D4C0 50                   <1>     push    eax
4142 0000D4C1 66B8A000             <1>     mov     ax, 80*2 ; 80 colums, 160 bytes per row
4143 0000D4C5 F7E1                 <1>     mul     ecx
4144                                <1>             ; eax = offset for start row number
4145 0000D4C7 01C6                 <1>     add     esi, eax
4146 0000D4C9 01C7                 <1>     add     edi, eax
4147 0000D4CB 58                   <1>     pop     eax
4148 0000D4CC 5A                   <1>     pop     edx
4149                                <1> sysvideo_4_1:
4150                                <1>     ;shl    ax, 1 ; * 2 ; convert start column number to offset
4151                                <1>     ; 23/07/2022
4152 0000D4CD D1E0                 <1>     shl     eax, 1
4153 0000D4CF 7404                 <1>     jz      short sysvideo_4_2
4154 0000D4D1 01C6                 <1>     add     esi, eax
4155 0000D4D3 01C7                 <1>     add     edi, eax
4156                                <1>             ; esi = source page window start offset
4157                                <1>             ; edi = destination page window start offset
4158                                <1> sysvideo_4_2:
4159 0000D4D5 59                   <1>     pop     ecx
4160                                <1>     ;mov    eax, 0B8000h+(80*25*2*8)
4161                                <1>     ; 21/11/2020
4162 0000D4D6 B800000C00           <1>     mov     eax, 0B8000h+(4096*8)
4163 0000D4DB 39C6                 <1>     cmp     esi, eax
4164                                <1>     ;jnb    sysret ; out of video page
4165                                <1>     ; 23/07/2022
4166 0000D4DD 7304                 <1>     jnb     short sysvideo_4_3  ; jmp sysret
4167 0000D4DF 39C6                 <1>     cmp     esi, eax
4168                                <1>     ;jnb    sysret  ;out of video page
4169                                <1>     ; 23/07/2022
4170 0000D4E1 7205                 <1>     jb      short sysvideo_4_4
4171                                <1> sysvideo_4_3:
4172 0000D4E3 E996F5FFFF           <1>     jmp     sysret
4173                                <1> sysvideo_4_4:
4174 0000D4E8 56                   <1>     push    esi ; ****
4175 0000D4E9 57                   <1>     push    edi ; ***
4176 0000D4EA 52                   <1>     push    edx ; **
4177 0000D4EB 51                   <1>     push    ecx ; *
4178 0000D4EC C1E910               <1>     shr     ecx, 16 ; top row
4179 0000D4EF C1EA10               <1>     shr     edx, 16 ; bottom row
4180                                <1>     ; 21/11/2020
4181                                <1>     ;cmp    ecx, 24 ; max. 25 rows
```

```
4182 0000D4F2 6683F918            <1>     cmp     cx, 24
4183 0000D4F6 7778                <1>     ja      short sysvideo_6 ; invalid, [u.r0] = 0
4184                              <1>     ;cmp    edx, 24 ; max. 25 rows
4185 0000D4F8 6683FA18            <1>     cmp     dx, 24
4186 0000D4FC 7772                <1>     ja      short sysvideo_6 ; invalid, [u.r0] = 0
4187 0000D4FE 28CA                <1>     sub     dl, cl ; end >= start
4188 0000D500 726E                <1>     jc      short sysvideo_6 ; invalid, [u.r0] = 0
4189                              <1>     ; 21/11/2020
4190 0000D502 89D3                <1>     mov     ebx, edx ; row count - 1
4191 0000D504 7415                <1>     jz      short sysvideo_4_5
4192 0000D506 50                  <1>     push    eax ; *****
4193 0000D507 B8A0000000          <1>     mov     eax, 80*2 ; bytes per row
4194 0000D50C F7E3                <1>     mul     ebx ; 21/11/2020
4195                              <1>             ; eax = window end offset
4196                              <1>             ; (for the last row, before adding column bytes)
4197 0000D50E 01C6                <1>     add     esi, eax
4198 0000D510 01C7                <1>     add     edi, eax
4199 0000D512 58                  <1>     pop     eax ; ***** ; mode 3 video memory end (0C000h)
4200 0000D513 39C6                <1>     cmp     esi, eax
4201                              <1>     ;ja     short sysvideo_6 ; invalid, [u.r0] = 0
4202 0000D515 7359                <1>     jnb     short sysvideo_6 ; 21/11/2020
4203 0000D517 39C7                <1>     cmp     edi, eax
4204                              <1>     ;ja     short sysvideo_6 ; invalid, [u.r0] = 0
4205 0000D519 7355                <1>     jnb     short sysvideo_6 ; 21/11/2020
4206                              <1> sysvideo_4_5:
4207 0000D51B 59                  <1>     pop     ecx ; *
4208 0000D51C 5A                  <1>     pop     edx ; **
4209 0000D51D 81E1FFFF0000        <1>     and     ecx, 0FFFFh
4210 0000D523 81E2FFFF0000        <1>     and     edx, 0FFFFh
4211                              <1>     ; 21/11/2020
4212                              <1>     ;cmp    ecx, 79 ; max. 80 columns
4213 0000D529 6683F94F            <1>     cmp     cx, 79
4214 0000D52D 7743                <1>     ja      short sysvideo_7 ; invalid, [u.r0] = 0
4215                              <1>     ;cmp    edx, 79 ; max. 80 columns
4216 0000D52F 6683FA4F            <1>     cmp     dx, 79
4217 0000D533 773D                <1>     ja      short sysvideo_7 ; invalid, [u.r0] = 0
4218 0000D535 28CA                <1>     sub     dl, cl
4219 0000D537 7639                <1>     jna     short sysvideo_7 ; invalid, [u.r0] = 0
4220                              <1>     ; 21/11/2020
4221 0000D539 740E                <1>     jz      short sysvideo_4_6
4222                              <1>     ; edx = column count (width) - 1
4223 0000D53B D0E2                <1>     shl     dl, 1 ; * 2 ; byte offset (in end row)
4224 0000D53D 01D6                <1>     add     esi, edx
4225 0000D53F 01D7                <1>     add     edi, edx
4226                              <1>             ; esi = source page window end offset
4227                              <1>             ; edi = destination page window end offset
4228 0000D541 39C6                <1>     cmp     esi, eax ; video memory end
4229                              <1>     ;ja     short sysvideo_7
4230 0000D543 732D                <1>     jnb     short sysvideo_7 ; 21/11/2020
4231 0000D545 39C7                <1>     cmp     edi, eax ; video memory end
4232                              <1>     ;ja     short sysvideo_7
4233 0000D547 7329                <1>     jnb     short sysvideo_7 ; 21/11/2020
4234                              <1> sysvideo_4_6:
4235 0000D549 5F                  <1>     pop     edi ; ***
4236 0000D54A 5E                  <1>     pop     esi ; ****
4237 0000D54B FEC3                <1>     inc     bl ; row count - 1 -> row count
4238 0000D54D FEC2                <1>     inc     dl ; column count
4239 0000D54F 88D7                <1>     mov     bh, dl
4240 0000D551 D0E2                <1>     shl     dl, 1 ; convert column count to byte offset
4241                              <1>     ; 21/11/2020
4242                              <1>     ; esi = source page window start offset
4243                              <1>     ; edi = destination page window start offset
4244 0000D553 B8A0000000          <1>     mov     eax, 80*2 ; bytes per row
4245                              <1>     ; Note: 160 bytes per row (even if move count < 160)
4246                              <1> sysvideo_5:
4247 0000D558 88F9                <1>     mov     cl, bh ; move/transfer -word- count per row
4248 0000D55A 0115[348E0100]      <1>     add     [u.r0], edx ; transfer count in bytes
4249 0000D560 F366A5              <1>     rep     movsw
4250 0000D563 01C6                <1>     add     esi, eax ; + 160 bytes to next row
4251 0000D565 01C7                <1>     add     edi, eax ; + 160 bytes to next row
4252 0000D567 FECB                <1>     dec     bl ; remain count of rows
4253 0000D569 75ED                <1>     jnz     short sysvideo_5
4254 0000D56B E90EF5FFFF          <1>     jmp     sysret
4255                              <1>
4256                              <1> sysvideo_6:
4257 0000D570 59                  <1>     pop     ecx ; *
4258 0000D571 5A                  <1>     pop     edx ; **
4259                              <1> sysvideo_7:
4260 0000D572 5F                  <1>     pop     edi ; ***
4261 0000D573 5E                  <1>     pop     esi ; ****
4262                              <1> sysvideo_8:
4263 0000D574 E905F5FFFF          <1>     jmp     sysret
4264                              <1>
4265                              <1> sysvideo_9:
4266                              <1>     ; user to system or system to user window transfer
4267                              <1>     ; 28/01/2021 (bl = 3 -> swap/exchange)
4268                              <1>     ;cmp    bl, 2
4269                              <1>     ;;ja    sysret  ; user to user transfer is invalid
4270                              <1>     ;               ; [u.r0] = 0
4271                              <1>     ;ja     short sysvideo_8 ; 26/12/2020
4272                              <1>
4273                              <1>     ; 28/01/2021
4274 0000D579 80FB02              <1>     cmp     bl, 2
4275 0000D57C 7604                <1>     jna     short sysvideo_9_8
4276                              <1>
4277                              <1>     ; swap/ exchange video memory and user mem windows
4278                              <1>     ; edi = swap address in user's memory space
4279 0000D57E 21FF                <1>     and     edi, edi
4280 0000D580 74F2                <1>     jz      short sysvideo_8 ; invalid ; 28/01/2021
4281                              <1>
4282                              <1> sysvideo_9_8:
4283 0000D582 56                  <1>     push    esi ; ****
4284 0000D583 57                  <1>     push    edi ; ***
4285 0000D584 52                  <1>     push    edx ; **
4286 0000D585 51                  <1>     push    ecx ; *
4287                              <1>
4288 0000D586 C1E910              <1>     shr     ecx, 16 ; top row
4289 0000D589 C1EA10              <1>     shr     edx, 16 ; bottom row
4290                              <1>
4291                              <1>     ; 21/11/2020
4292                              <1>     ;cmp    ecx, 24 ; max. 25 rows
4293 0000D58C 6683F918            <1>     cmp     cx, 24
4294 0000D590 77DE                <1>     ja      short sysvideo_6 ; invalid, [u.r0] = 0
4295                              <1>     ;cmp    edx, 24 ; max. 25 rows
4296 0000D592 6683FA18            <1>     cmp     dx, 24
4297 0000D596 77D8                <1>     ja      short sysvideo_6 ; invalid, [u.r0] = 0
4298 0000D598 28CA                <1>     sub     dl, cl
4299 0000D59A 72D4                <1>     jc      short sysvideo_6 ; invalid, [u.r0] = 0
4300                              <1>
4301                              <1>     ;mov    ch, cl ; top row
4302                              <1>     ;mov    cl, [ACTIVE_PAGE]
4303                              <1>
4304                              <1>     ;mov    edi, 80*25*2 ; 4000
4305                              <1>     ; 21/11/2020
```

```
4306                                    <1>     ;mov    edi, 4096 ; [CRT_LEN = 4096 for video mode 3
4307                                    <1>     ;shl    edi, cl  ; ! wrong for page 2 to page 7 !
4308                                    <1>     ;;add   edi, 0B8000h - 80*25*2
4309                                    <1>     ;add    edi, 0B8000h - 1000h ; - 4096
4310                                    <1>
4311                                    <1>     ; 21/11/2020
4312                                    <1>     ;xor    eax, eax
4313                                    <1>     ;mov    edi, 0B8000h
4314                                    <1>     ;and    cl, cl ; is video page = 0 ?
4315                                    <1>     ;jz     short sysvideo_9_1 ; yes
4316                                    <1>     ; eax = 0
4317                                    <1>
4318                                    <1> ;sysvideo_9_0:
4319                                    <1>     ;add    ax, 4096
4320                                    <1>     ;dec    cl
4321                                    <1>     ;jnz    short sysvideo_9_0
4322                                    <1>     ;add    edi, eax
4323                                    <1>     ;       ; edi = video page start address
4324                                    <1>
4325                                    <1>     ; 21/11/2020
4326 0000D59C BF00800B00               <1>     mov     edi, 0B8000h
4327 0000D5A1 803D[AE760100]00         <1>     cmp     byte [ACTIVE_PAGE], 0
4328 0000D5A8 760C                     <1>     jna     short sysvideo_9_1
4329                                    <1> stsvideo_9_0:
4330 0000D5AA B010                     <1>     mov     al, 16 ; 4096/256
4331 0000D5AC F625[AE760100]           <1>     mul     byte [ACTIVE_PAGE]
4332 0000D5B2 86C4                     <1>     xchg    ah, al ; * 256
4333 0000D5B4 01C7                     <1>     add     edi, eax
4334                                    <1>             ; edi = video page start address
4335                                    <1> sysvideo_9_1:
4336                                    <1>     ; bl = transfer direction
4337                                    <1>     ;     (1 = from user, 2 = to user)
4338                                    <1>     ;     (3 = swap) ; 28/01/2021
4339 0000D5B6 88D7                     <1>     mov     bh, dl ; row count - 1
4340                                    <1>     ;mov    dl, ch ; top row
4341                                    <1>     ; 21/11/2020
4342 0000D5B8 08C9                     <1>     or      cl, cl ; top row number
4343 0000D5BA 7408                     <1>     jz      short sysvideo_9_2
4344                                    <1>
4345                                    <1>     ;mov    eax, 80*2
4346 0000D5BC 66B8A000                 <1>     mov     ax, 80*2 ; 160, bytes per row
4347 0000D5C0 F7E1                     <1>     mul     ecx ; 22/11/2020
4348 0000D5C2 01C7                     <1>     add     edi, eax
4349                                    <1>             ; edi = window start address for top row
4350                                    <1> sysvideo_9_2:
4351 0000D5C4 59                       <1>     pop     ecx ; *
4352 0000D5C5 5A                       <1>     pop     edx ; **
4353 0000D5C6 81E1FFFF0000             <1>     and     ecx, 0FFFFh
4354 0000D5CC 81E2FFFF0000             <1>     and     edx, 0FFFFh
4355                                    <1>     ; 21/11/2020
4356                                    <1>     ;cmp    ecx, 79 ; max. 80 columns
4357 0000D5D2 6683F94F                 <1>     cmp     cx, 79
4358 0000D5D6 779A                     <1>     ja      short sysvideo_7 ; invalid, [u.r0] = 0
4359                                    <1>     ;cmp    edx, 79 ; max. 80 columns
4360 0000D5D8 6683FA4F                 <1>     cmp     dx, 79
4361 0000D5DC 7794                     <1>     ja      short sysvideo_7 ; invalid, [u.r0] = 0
4362                                    <1>
4363 0000D5DE 28CA                     <1>     sub     dl, cl
4364 0000D5E0 7290                     <1>     jc      short sysvideo_7 ; invalid, [u.r0] = 0
4365                                    <1>
4366 0000D5E2 08C9                     <1>     or      cl, cl ; left column
4367 0000D5E4 7404                     <1>     jz      short sysvideo_9_3 ; 0
4368                                    <1>
4369                                    <1>     ; 21/11/2020
4370 0000D5E6 D0E1                     <1>     shl     cl, 1  ; column * 2
4371 0000D5E8 01CF                     <1>     add     edi, ecx
4372                                    <1>             ; edi = window start addr for top left column
4373                                    <1> sysvideo_9_3:
4374 0000D5EA 88D1                     <1>     mov     cl, dl
4375 0000D5EC FEC1                     <1>     inc     cl ; column count
4376 0000D5EE D0E1                     <1>     shl     cl, 1 ; column count * 2
4377                                    <1>             ; ecx = transfer count per row
4378                                    <1>
4379 0000D5F0 58                       <1>     pop     eax ; *** (swap address)
4380 0000D5F1 5E                       <1>     pop     esi ; ****
4381                                    <1>
4382 0000D5F2 FEC7                     <1>     inc     bh  ; row count
4383                                    <1>
4384                                    <1>     ;mov    edx, 80*2
4385 0000D5F4 B2A0                     <1>     mov     dl, 80*2  ; bytes per row
4386                                    <1>     ;
4387                                    <1>     ;cmp    bl, 1 ; transfer direction
4388                                    <1>     ;ja     short sysvideo_11 ; system to user transfer
4389                                    <1>     ; 28/01/2021
4390 0000D5F6 F6C301                   <1>     test    bl, 1
4391 0000D5F9 7439                     <1>     jz      short sysvideo_11 ; system to user transfer
4392                                    <1>
4393                                    <1>     ; user to system video/display page window transfer (mode 0)
4394 0000D5FB 21C0                     <1>     and     eax, eax ; swap address
4395 0000D5FD 741B                     <1>     jz      short sysvideo_10 ; no window swap
4396                                    <1> sysvideo_9_7: ; 28/01/2021
4397                                    <1>     ; save previous window content in user's buffer (swap address)
4398 0000D5FF 56                       <1>     push    esi ; user buffer
4399 0000D600 57                       <1>     push    edi ; beginning address of the window
4400                                    <1>     ; 21/11/2020
4401 0000D601 53                       <1>     push    ebx ; save bh
4402 0000D602 89FE                     <1>     mov     esi, edi
4403 0000D604 89C7                     <1>     mov     edi, eax
4404                                    <1> sysvideo_9_4:
4405 0000D606 E8DF340000               <1>     call    transfer_to_user_buffer ; fast transfer
4406 0000D60B 7208                     <1>     jc      short sysvideo_9_5
4407                                    <1>     ; ecx = actual transfer count (must be same with input)
4408 0000D60D 01D6                     <1>     add     esi, edx ; next row address of (video page) window
4409 0000D60F 01CF                     <1>     add     edi, ecx ; next row address of user's window
4410                                    <1>             ; Note: ecx may be less than row length of video page
4411                                    <1>             ; user's window uses offset according to window width
4412 0000D611 FECF                     <1>     dec     bh
4413 0000D613 75F1                     <1>     jnz     short sysvideo_9_4 ; repeat for next row
4414                                    <1> sysvideo_9_5:
4415 0000D615 5B                       <1>     pop     ebx ; restore bh
4416 0000D616 5F                       <1>     pop     edi
4417 0000D617 5E                       <1>     pop     esi
4418                                    <1>     ;jnc    short sysvideo_10
4419 0000D618 7215                     <1>     jc      short sysvideo_9_6 ; 28/01/2021
4420                                    <1> ;sysvideo_9_6:
4421                                    <1> ;   jmp     sysret ; [u.r0] = 0
4422                                    <1>
4423                                    <1> sysvideo_10:
4424                                    <1>     ; user to system video/display page window transfer (mode 0)
4425                                    <1>     ; esi = user buffer
4426 0000D61A E815350000               <1>     call    transfer_from_user_buffer ; fast transfer
4427                                    <1>     ;jc     sysret
4428 0000D61F 720E                     <1>     jc      short sysvideo_9_6 ; 28/01/2021
4429                                    <1>     ; ecx = actual transfer count (must be same with input)
```

```
4430 0000D621 010D[348E0100]      <1>     add     [u.r0], ecx ; actual transfer count
4431 0000D627 01D7                <1>     add     edi, edx ; next row address of (video page) window
4432 0000D629 01CE                <1>     add     esi, ecx ; next row address of user's window
4433                              <1>             ; Note: ecx may be less than row length of video page
4434                              <1>             ; user's window uses offset according to window width
4435 0000D62B FECF                <1>     dec     bh
4436 0000D62D 75EB                <1>     jnz     short sysvideo_10 ; repeat for next row
4437                              <1>     ;jmp    sysret
4438                              <1> sysvideo_9_6:
4439 0000D62F E94AF4FFFF          <1>     jmp     sysret
4440                              <1>
4441                              <1> sysvideo_11:
4442                              <1>     ; system to user video/display page window transfer (mode 0)
4443 0000D634 87F7                <1>     xchg    edi, esi
4444                              <1> sysvideo_12:
4445                              <1>     ; esi = beginning addr of the (screen, video page) window
4446                              <1>     ; edi =user's buffer
4447 0000D636 E8AF340000          <1>     call    transfer_to_user_buffer ; fast transfer
4448                              <1>     ;jc     sysret
4449                              <1>     ; 23/07/2022
4450 0000D63B 72F2                <1>     jc      short sysvideo_9_6 ; jmp sysret
4451                              <1>
4452                              <1>     ; ecx = actual transfer count (must be same with input)
4453 0000D63D 010D[348E0100]      <1>     add     [u.r0], ecx
4454 0000D643 01D6                <1>     add     esi, edx ; next row (edx = 160)
4455 0000D645 01CF                <1>     add     edi, ecx ; next row of the user's window
4456                              <1>                      ; (ecx <= 160)
4457 0000D647 FECF                <1>     dec     bh
4458 0000D649 75EB                <1>     jnz     short sysvideo_12
4459                              <1> sysvideo_12_0:
4460 0000D64B E92EF4FFFF          <1>     jmp     sysret
4461                              <1>
4462                              <1> sysvideo_13:
4463                              <1>     ; 28/12/2020
4464 0000D650 80FF01              <1>     cmp     bh, 1
4465 0000D653 7752                <1>         ja      short sysvideo_15 ; 23/11/2020
4466                              <1>
4467                              <1>     ; 25/02/2021
4468                              <1>     ; 12/02/2021
4469                              <1>     ; 29/01/2021, 31/01/2021
4470                              <1>     ; 23/11/2020 (TRDOS 386 v2.0.3)
4471                              <1>     ; (major modification, from mode 13h to all VGA modes,
4472                              <1>     ;  except super VGA modes and liner frame buffer method)
4473                              <1>
4474                              <1>     ; BH = 1 = VGA Graphics mode (0A0000h) data transfers
4475                              <1>
4476                              <1>     ; 29/01/2021
4477 0000D655 66B84001            <1>     mov     ax, 320 ; 320 pixels
4478 0000D659 F6C380              <1>     test    bl, 80h ; bit 7 (screen width, 640 pixels)
4479 0000D65C 7407                <1>     jz      short sysvideo_13_0
4480                              <1>     ;shl    ax, 1   ; 640 pixels
4481                              <1>     ; 23/07/2022
4482 0000D65E D1E0                <1>     shl     eax, 1
4483                              <1>     ;
4484 0000D660 80E37F              <1>     and     bl, 7Fh
4485 0000D663 7405                <1>     jz      short sysvideo_14
4486                              <1> sysvideo_13_0:
4487                              <1>     ; 29/01/2021
4488 0000D665 80FB41              <1>     cmp     bl, 41h
4489 0000D668 77E1                <1>     ja      short sysvideo_12_0 ; invalid (unknown) sub function
4490                              <1> sysvideo_14:
4491 0000D66A 66A3[569D0100]      <1>     mov     [v_width], ax  ; save screen width
4492 0000D670 C705[5A9D0100]0000- <1>     mov     dword [v_mem], 0A0000h  ; save video memory address
4492 0000D678 0A00                <1>
4493 0000D67A C705[5E9D0100]0000- <1>     mov     dword [v_siz], 65536  ; save video memory size
4493 0000D682 0100                <1>
4494 0000D684 C705[669D0100]0000- <1>     mov     dword [v_end], 0B0000h; save end of video page
4494 0000D68C 0B00                <1>
4495 0000D68E B708                <1>     mov     bh, 8
4496 0000D690 883D[599D0100]      <1>     mov     [v_bpp], bh ; 8       ; bits per pixel (256 colors)
4497 0000D696 881D[589D0100]      <1>     mov     [v_ops], bl    ; VGA data transfer options
4498                              <1>     ;mov    [maskbuff], edi ; 25/02/2021
4499 0000D69C 893D[6A9D0100]      <1>     mov     [maskcolor], edi ; 25/02/2021
4500                              <1>                     ; save mask color or bitmask buffer address
4501 0000D6A2 E9BE000000          <1>     jmp     sysvideo_15_7
4502                              <1>
4503                              <1> sysvideo_15:
4504                              <1>     ; 23/07/2022
4505                              <1>     ; 28/12/2020
4506 0000D6A7 80FF02              <1>     cmp     bh, 2
4507                              <1>         ;ja sysvideo_16
4508                              <1>     ; 23/07/2022
4509 0000D6AA 7605                <1>     jna     short sysvideo_15_17
4510 0000D6AC E9ED1E0000          <1>     jmp     sysvideo_16
4511                              <1> sysvideo_15_17:    ; 23/07/2022
4512                              <1>     ; 25/02/2021
4513                              <1>     ; 12/02/2021
4514                              <1>     ; 30/01/2021 - 31/01/2021
4515                              <1>     ; 01/01/2021 - 29/01/2021
4516                              <1>     ; 26/12/2020 - 27/12/2020
4517                              <1>     ; 25/12/2020 (TRDOS 386 v2.0.3)
4518                              <1>     ;
4519                              <1>     ; BH = 2 = SVGA (VESA VBE) Graphics mode (LFB) data transfers
4520                              <1>
4521                              <1>     ; 25/12/2020
4522                              <1>     ; resolution table entry will be saved into EBP register
4523                              <1>
4524 0000D6B1 803D[86090000]02    <1>     cmp     byte [vbe3], 2 ; VESA VBE 3 video bios
4525                              <1>                            ; or BOCHS/QEMU/VIRTUALBOX emu video bios
4526 0000D6B8 724E                <1>     jb      short sysvideo_15_4 ; no, nothing to do !
4527 0000D6BA 770B                <1>     ja      short sysvideo_15_0 ; yes
4528                              <1>
4529                              <1>     ; Only Bochs/Plex86 (emu) vbe2 video bios is usable in pmid
4530                              <1>     ; (if [vbe3] = 2)
4531 0000D6BC A0[87090000]        <1>     mov     al, [vbe2bios] ; Bochs vbios sign is from C0h to C5h
4532 0000D6C1 24F0                <1>     and     al, 0F0h
4533 0000D6C3 3CC0                <1>     cmp     al, 0C0h
4534 0000D6C5 7541                <1>     jne     short sysvideo_15_4 ; unknown (vbe2) video bios
4535                              <1> sysvideo_15_0:
4536                              <1>     ; 29/01/2021
4537 0000D6C7 80FB41              <1>     cmp     bl, 41h
4538 0000D6CA 773C                <1>     ja      short sysvideo_15_4 ; invalid (unknown) sub function
4539                              <1>     ; 29/01/2021
4540 0000D6CC 881D[589D0100]      <1>     mov     [v_ops], bl    ; SVGA data transfer options
4541                              <1>
4542 0000D6D2 89D8                <1>     mov     eax, ebx ; hw of ebx is vesa vbe video mode
4543 0000D6D4 C1E810              <1>     shr     eax, 16 ; ax = vesa vbe video mode
4544 0000D6D7 7513                <1>     jnz     short sysvideo_15_2
4545                              <1>     ; ax = 0
4546                              <1>
4547                              <1>     ; check & use current video mode
4548 0000D6D9 803D[1E680000]FF    <1>     cmp     byte [CRT_MODE], 0FFh ; extended (SVGA) mode ?
4549 0000D6E0 7526                <1>     jne     short sysvideo_15_4 ; no
4550                              <1> sysvideo_15_1:
```

```
                                <1>     ; use current vbe (svga) video mode
0000D6E2 66A1[EE9C0100]         <1>     mov     ax, [video_mode] ; extended (SVGA, VESA VBE) mode
0000D6E8 6625FF01               <1>     and     ax, 1FFh ; vesa vbe video mode: 1XXh
                                <1> sysvideo_15_2:
                                <1>     ; 29/01/2021
                                <1>     ;mov    [maskbuff], edi ; 25/02/2021
0000D6EC 893D[6A9D0100]         <1>     mov     [maskcolor], edi ; 25/02/2021
                                <1>                     ; save mask color or bitmask buffer address
0000D6F2 BD[AA6B0000]           <1>     mov     ebp, b_vbe_modes ; vbe mode table (in 'vidata.s')
                                <1> sysvideo_15_3:
0000D6F7 663B4500               <1>     cmp     ax, [ebp]
0000D6FB 7410                   <1>     je      short sysvideo_15_5
0000D6FD 83C508                 <1>     add     ebp, 8 ; vbe mode table entry size
0000D700 81FD[6A6C0000]         <1>     cmp     ebp, end_of_b_vbe_modes
0000D706 72EF                   <1>     jb      short sysvideo_15_3
                                <1> sysvideo_15_4:
                                <1>     ; desired video mode is not a valid (implemented)
                                <1>     ;         extended (VESA VBE, SVGA) video mode
                                <1>     ;
                                <1>     ; nothing to do !
                                <1>
                                <1>     ; [u.r0] = 0  ; return value of EAX
0000D708 E971F3FFFF             <1>     jmp     sysret
                                <1>
                                <1> sysvideo_15_5:
                                <1>     ; get LFB address
0000D70D A1[FC9C0100]           <1>     mov     eax, [LFB_ADDR] ; [LFB_Info+LFBINFO.LFB_addr]
0000D712 09C0                   <1>     or      eax, eax
0000D714 7509                   <1>     jnz     short sysvideo_15_6
0000D716 66A1[1A0F0000]         <1>     mov     ax, [def_LFB_addr] ; default LFB addr
                                <1>                                ; (for vbe mode 118h)
0000D71C C1E010                 <1>     shl     eax, 16
                                <1>     ; 27/12/2020
                                <1>     ;jz     short sysvideo_15_4
                                <1> sysvideo_15_6:
                                <1>     ; 29/01/2021
0000D71F A3[5A9D0100]           <1>     mov     [v_mem], eax ; save video memory address
                                <1>
                                <1>     ; 27/12/2020
                                <1>     ; 26/12/2020
0000D724 8B4502                 <1>     mov     eax, [ebp+2] ; width, height
                                <1>     ; 29/01/2021
0000D727 66A3[569D0100]         <1>     mov     [v_width], ax ; save screen width
                                <1>     ; 28/12/2020
0000D72D 8A7D06                 <1>     mov     bh, [ebp+6] ; bpp
                                <1>     ; 28/02/2021
                                <1>     ; check default truecolor bpp value and use
                                <1>     ; 32bpp instead of 24bpp if the default value
                                <1>     ; has been set to 32bpp.
0000D730 80FF18                 <1>     cmp     bh, 24
0000D733 750B                   <1>     jne     short sysvideo_15_16
0000D735 803D[43730100]20       <1>     cmp     byte [truecolor], 32
                                <1>             ; Default truecolor bpp value,
                                <1>             ; it is 32 for VBE3 video bios
                                <1>             ; (it can be set to 32 or 24)
0000D73C 7502                   <1>     jne     short sysvideo_15_16 ; not VBE3 !
                                <1>                             ; or it is set to 24
0000D73E B720                   <1>     mov     bh, 32
                                <1>     ; 28/02/2021
                                <1> sysvideo_15_16:
                                <1>     ; 29/01/2021
0000D740 883D[599D0100]         <1>     mov     [v_bpp], bh ; bits per pixel
                                <1>
0000D746 52                     <1>     push    edx ; *
0000D747 0FB7D0                 <1>     movzx   edx, ax  ; width
0000D74A C1E810                 <1>     shr     eax, 16  ; height
0000D74D F7E2                   <1>     mul     edx
                                <1>     ; eax = linear frame buffer size (pixels)
                                <1>     ; 29/01/2021
0000D74F A3[5E9D0100]           <1>     mov     [v_siz], eax ; save video page size
0000D754 E8FD000000             <1>     call    pixels_to_byte_count
0000D759 0305[5A9D0100]         <1>     add     eax, [v_mem]
0000D75F A3[669D0100]           <1>     mov     [v_end], eax ; save end of video page
0000D764 5A                     <1>     pop     edx ; *
                                <1>
                                <1>     ; bh = bits per pixel
                                <1>     ; (bh will not be used after here, 29/01/2021)
                                <1>
                                <1>     ; bl = pixel operations & options
                                <1>     ; ecx, edx, esi, edi input parameters
                                <1>     ; [maskcolor] = edi input ; 25/02/2021
                                <1>
                                <1> sysvideo_15_7:
                                <1>     ; 29/01/2021
                                <1>     ;test   byte [v_ops], 40h  ; system to user ?
0000D765 F6C340                 <1>     test    bl, 40h
0000D768 7517                   <1>     jnz     short sysvideo_15_9
                                <1>
0000D76A 31C0                   <1>     xor     eax, eax
0000D76C 88D8                   <1>     mov     al, bl
0000D76E BB[94D80000]           <1>     mov     ebx, pixel_ops
0000D773 240F                   <1>     and     al, 0Fh ; isolate 16 pixel operations
0000D775 C0E002                 <1>     shl     al, 2 ; * 4 for dword table pointers
0000D778 01C3                   <1>     add     ebx, eax
                                <1>
                                <1>     ; ebx = subroutine address
                                <1>
                                <1>     ; ecx, edx, esi, edi input parameters
                                <1>     ; [maskbuff] = edi input
                                <1>     ; [maskcolor] = edi input ; 25/02/2021
                                <1>
0000D77A FF13                   <1>     call    [ebx]
                                <1> sysvideo_15_8:
0000D77C E9FDF2FFFF             <1>     jmp     sysret
                                <1>
                                <1> sysvideo_15_9:
                                <1>     ; system to user display page or window copy
                                <1>     ;test   byte [v_ops], 1 ; window copy ?
0000D781 F6C301                 <1>     test    bl, 1
0000D784 7521                   <1>     jnz     short sysvideo_15_10
                                <1>
                                <1>     ; display page (full screen copy)
0000D786 8B35[5A9D0100]         <1>     mov     esi, [v_mem] ; LFB start address
0000D78C A1[5E9D0100]           <1>     mov     eax, [v_siz]
0000D791 E8C0000000             <1>     call    pixels_to_byte_count
0000D796 89C1                   <1>     mov     ecx, eax ; transfer count in bytes
                                <1>     ;edi = user's buffer address
0000D798 E84D330000             <1>     call    transfer_to_user_buffer
0000D79D 72DD                   <1>     jc      short sysvideo_15_8
0000D79F 890D[348E0100]         <1>     mov     [u.r0], ecx
0000D7A5 EBD5                   <1>     jmp     short sysvideo_15_8
                                <1>
                                <1> sysvideo_15_10:
0000D7A7 E820000000             <1>     call    sysvideo_15_12 ; window preparations
```

```
4675 0000D7AC 72CE                <1>     jc      short sysvideo_15_8
4676                              <1>
4677 0000D7AE 8B35[629D0100]      <1>     mov     esi, [v_str]
4678                              <1> sysvideo_15_11:
4679                              <1>     ; esi = window's current row address (video mem)
4680                              <1>     ; edi = current row (virtual) addr in user's buff
4681                              <1>     ; ecx = transfer count per row
4682 0000D7B4 E831330000          <1>     call    transfer_to_user_buffer
4683 0000D7B9 72C1                <1>     jc      short sysvideo_15_8
4684 0000D7BB 010D[348E0100]      <1>     add     [u.r0], ecx
4685 0000D7C1 4B                  <1>     dec     ebx
4686 0000D7C2 74B8                <1>     jz      short sysvideo_15_8 ; ok.
4687                              <1>     ; next row
4688 0000D7C4 01CF                <1>     add     edi, ecx ; next row in user's buffer
4689 0000D7C6 01D6                <1>     add     esi, edx ; next row of window (system)
4690 0000D7C8 EBEA                <1>     jmp     short sysvideo_15_11
4691                              <1>
4692                              <1> sysvideo_15_14:
4693 0000D7CA F9                  <1>     stc     ; error !
4694                              <1> sysvideo_15_15:
4695 0000D7CB C3                  <1>     retn
4696                              <1>
4697                              <1> sysvideo_15_12:
4698                              <1>     ; 30/01/2021
4699                              <1>     ; 29/01/2021
4700                              <1>     ; Window address preparations for window copy
4701 0000D7CC 6621D2              <1>     and     dx, dx
4702 0000D7CF 74F9                <1>     jz      short sysvideo_15_14 ; invalid (zero columns)
4703                              <1>     ;test   edx, 0FFFF0000h
4704                              <1>     ;jz     short sysvideo_15_14 ; invalid (zero rows)
4705 0000D7D1 81FA00000100        <1>     cmp     edx, 65536
4706 0000D7D7 72F2                <1>     jb      short sysvideo_15_15 ; invalid (zero rows)
4707 0000D7D9 89C8                <1>     mov     eax, ecx ; start position (row, column)
4708 0000D7DB E899000000          <1>     call    calc_pixel_offset
4709 0000D7E0 3B05[5E9D0100]      <1>     cmp     eax, [v_siz]
4710 0000D7E6 73E2                <1>     jnb     short sysvideo_15_14 ; out of display page
4711                              <1>                                  ; nothing to do
4712 0000D7E8 E869000000          <1>     call    pixels_to_byte_count
4713 0000D7ED 0305[5A9D0100]      <1>     add     eax, [v_mem]
4714 0000D7F3 A3[629D0100]        <1>     mov     [v_str], eax ; window start address
4715                              <1>                          ; (addr of top left corner)
4716                              <1>     ; check column limit
4717 0000D7F8 89C8                <1>     mov     eax, ecx
4718 0000D7FA 6601D0              <1>     add     ax, dx  ; add columns to start column
4719 0000D7FD 72CC                <1>     jc      short sysvideo_15_15 ; cf = 1
4720 0000D7FF 663B05[569D0100]    <1>     cmp     ax, [v_width]
4721 0000D806 77C2                <1>     ja      short sysvideo_15_14
4722                              <1>
4723 0000D808 89D0                <1>     mov     eax, edx ; size
4724 0000D80A 2D00000100          <1>     sub     eax, 65536 ; row count -> 0 based row #
4725 0000D80F E865000000          <1>     call    calc_pixel_offset
4726 0000D814 3B05[5E9D0100]      <1>     cmp     eax, [v_siz] ; video (display) page size
4727 0000D81A 77AE                <1>     ja      short sysvideo_15_14 ; out of display page
4728                              <1>                                  ; nothing to do
4729 0000D81C E835000000          <1>     call    pixels_to_byte_count
4730 0000D821 0305[629D0100]      <1>     add     eax, [v_str] ; window start address
4731 0000D827 3B05[669D0100]      <1>     cmp     eax, [v_end] ; window end address (+1)
4732                              <1>                          ; (addr of bottom right corner +1)
4733 0000D82D 779B                <1>     ja      short sysvideo_15_14 ; out of display page
4734                              <1>                                  ; nothing to do
4735 0000D82F 89D3                <1>     mov     ebx, edx
4736 0000D831 C1EB10              <1>     shr     ebx, 16
4737                              <1>     ; ebx = row count
4738 0000D834 81E2FFFF0000        <1>     and     edx, 0FFFFh
4739                              <1>     ; edx = transfer count per row (from user's buffer)
4740                              <1>     ;       (in pixels, window width)
4741 0000D83A 89D0                <1>     mov     eax, edx
4742 0000D83C A3[6E9D0100]        <1>     mov     [pixcount], eax ; 27/02/2021
4743 0000D841 E810000000          <1>     call    pixels_to_byte_count
4744 0000D846 89C1                <1>     mov     ecx, eax
4745                              <1>     ; ecx = transfer count per row (from user's buffer)
4746                              <1>     ;       (in bytes, window width)
4747 0000D848 66A1[569D0100]      <1>     mov     ax, [v_width]
4748 0000D84E E803000000          <1>     call    pixels_to_byte_count
4749 0000D853 89C2                <1>     mov     edx, eax
4750                              <1>     ; edx = byte count per row
4751 0000D855 C3                  <1>     retn ; cf = 0
4752                              <1>
4753                              <1> pixels_to_byte_count:
4754                              <1>     ; 29/01/2021
4755                              <1>     ; INPUT:
4756                              <1>     ;   eax = pixel count
4757                              <1>     ; OUTPUT:
4758                              <1>     ;   eax = byte count
4759                              <1>     ;
4760 0000D856 803D[599D0100]08    <1>     cmp     byte [v_bpp], 8
4761 0000D85D 7619                <1>     jna     short pixtobc_3 ; 8 bit colors
4762 0000D85F 803D[599D0100]18    <1>     cmp     byte [v_bpp], 24
4763 0000D866 720A                <1>     jb      short pixtobc_1 ; 16 bit colors
4764 0000D868 770B                <1>     ja      short pixtobc_2 ; 32 bit colors
4765                              <1>     ; 24 bit pixels
4766                              <1>     ; eax = eax * 3
4767                              <1>     ;push   edx
4768                              <1>     ;mov    edx, eax
4769                              <1>     ;shl    eax, 1
4770                              <1>     ;add    eax, edx
4771                              <1>     ;pop    edx
4772 0000D86A 50                  <1>     push    eax
4773 0000D86B D1E0                <1>     shl     eax, 1
4774 0000D86D 010424              <1>     add     [esp], eax
4775 0000D870 58                  <1>     pop     eax
4776 0000D871 C3                  <1>     retn
4777                              <1> pixtobc_1:
4778                              <1>     ; 32 bit pixels
4779                              <1>     ; eax = eax * 2
4780 0000D872 D1E0                <1>     shl     eax, 1
4781 0000D874 C3                  <1>     retn
4782                              <1> pixtobc_2:
4783                              <1>     ; 16 bit pixels
4784                              <1>     ; eax = eax * 4
4785 0000D875 C1E002              <1>     shl     eax, 2
4786                              <1> pixtobc_3:
4787 0000D878 C3                  <1>     retn
4788                              <1>
4789                              <1> calc_pixel_offset:
4790                              <1>     ; 29/01/2021
4791                              <1>     ; INPUT:
4792                              <1>     ;   eax = pixel position (row, column)
4793                              <1>     ; OUTPUT:
4794                              <1>     ;   eax = pixel offset (linear address)
4795                              <1>     ;
4796 0000D879 52                  <1>     push    edx
4797 0000D87A 50                  <1>     push    eax
4798 0000D87B C1E810              <1>     shr     eax, 16
```

```
4799 0000D87E 7409                <1>     jz      short cpixo_0
4800                              <1>     ; eax = row
4801 0000D880 0FB715[569D0100]    <1>     movzx   edx, word [v_width]
4802 0000D887 F7E2                <1>     mul     edx
4803                              <1> cpixo_0:
4804                              <1>     ; eax = row * screen width
4805 0000D889 5A                  <1>     pop     edx
4806 0000D88A 81E2FFFF0000        <1>     and     edx, 0FFFFh
4807                              <1>     ; edx = column
4808 0000D890 01D0                <1>     add     eax, edx
4809                              <1>     ; eax = (row * screen width) + column
4810 0000D892 5A                  <1>     pop     edx
4811 0000D893 C3                  <1>     retn
4812                              <1>
4813                              <1>     ; 02/02/2021
4814                              <1>     ; 29/01/2021
4815                              <1> pixel_ops:
4816 0000D894 [D4D80000]          <1>     dd      pix_op_cpy ; copy pixels (user to system)
4817 0000D898 [1FDE0000]          <1>     dd      pix_op_new ; change (new, fill) color
4818 0000D89C [3CD90000]          <1>     dd      pix_op_add ; add color (up to 0FFh)
4819 0000D8A0 [EED90000]          <1>     dd      pix_op_sub ; sub color (down to 0)
4820 0000D8A4 [09DC0000]          <1>     dd      pix_op_orc ; or color
4821 0000D8A8 [BBDC0000]          <1>     dd      pix_op_and ; and color
4822 0000D8AC [6DDD0000]          <1>     dd      pix_op_xor ; xor color
4823 0000D8B0 [E3DE0000]          <1>     dd      pix_op_not ; not color
4824 0000D8B4 [8DDF0000]          <1>     dd      pix_op_neg ; neg color
4825 0000D8B8 [37E00000]          <1>     dd      pix_op_inc ; inc color
4826 0000D8BC [E1E00000]          <1>     dd      pix_op_dec ; dec color
4827 0000D8C0 [A0DA0000]          <1>     dd      pix_op_mix ; mix color
4828 0000D8C4 [67DB0000]          <1>     dd      pix_op_rpl ; replace color
4829 0000D8C8 [8BE10000]          <1>     dd      pix_op_blk ; copy pixel block(s) (sys)
4830 0000D8CC [39E20000]          <1>     dd      pix_op_lin ; write line(s)
4831 0000D8D0 [1CE60000]          <1>     dd      pix_op_chr ; write character (font)
4832                              <1>
4833                              <1> pix_op_cpy:
4834                              <1>     ; 21/02/2021
4835                              <1>     ; 06/02/2021
4836                              <1>     ; 30/01/2021
4837                              <1>     ; COPY PIXELS
4838                              <1>     ;
4839                              <1>     ; INPUT:
4840                              <1>     ;  If bit 4 of BL or [v_ops] = 1 -window copy-
4841                              <1>     ;  ECX = start position (row, column)
4842                              <1>     ;         (HW = row, CX = column)
4843                              <1>     ;  EDX = size (rows, colums)
4844                              <1>     ;         (HW = rows, DX = columns)
4845                              <1>     ;        (0 -> invalid
4846                              <1>     ;         (1 -> horizontal or vertical line)
4847                              <1>     ;  If bit 4 of BL or [v_ops] = 0 -full screen-
4848                              <1>     ;     ECX and EDX will not be used
4849                              <1>     ;  ESI = user's buffer address
4850                              <1>     ;  [maskcolor] = mask color (to be excluded)
4851                              <1>     ;
4852                              <1>     ; OUTPUT:
4853                              <1>     ;       [u.r0] will be > 0 if succesful
4854                              <1>
4855 0000D8D4 F605[589D0100]10    <1>     test    byte [v_ops], 10h ; display page or window ?
4856 0000D8DB 752E                <1>     jnz     short pix_op_cpy_w ; window
4857                              <1>
4858 0000D8DD 8B3D[5A9D0100]      <1>     mov     edi, [v_mem] ; 21/02/2021
4859                              <1>
4860                              <1>     ; Copy user's buffer content do display page
4861                              <1>     ; (full screen copy)
4862 0000D8E3 A1[5E9D0100]        <1>     mov     eax, [v_siz] ; video page size
4863 0000D8E8 E869FFFFFF          <1>     call    pixels_to_byte_count
4864 0000D8ED 89C1                <1>     mov     ecx, eax ; transfer count
4865                              <1>     ; esi = user's buffer address (virtual)
4866 0000D8EF F605[589D0100]20    <1>     test    byte [v_ops], 20h ; masked copy ?
4867 0000D8F6 7405                <1>     jz      short pix_op_cpy_0 ; no
4868 0000D8F8 E96F0F0000          <1>     jmp     m_pix_op_cpy ; copy pixels except mask color
4869                              <1> pix_op_cpy_0:
4870                              <1>     ; esi = user buffer for full screen copy
4871                              <1>     ; edi = start of video memory
4872                              <1>     ;       (start of display page)
4873                              <1>     ; ecx = byte count (display page size in bytes)
4874 0000D8FD E832320000          <1>     call    transfer_from_user_buffer
4875 0000D902 7206                <1>     jc      short pix_op_cpy_1
4876 0000D904 890D[348E0100]      <1>     mov     [u.r0], ecx
4877                              <1> pix_op_cpy_1:
4878 0000D90A C3                  <1>     retn    ; 06/02/2021
4879                              <1>
4880                              <1> pix_op_cpy_w:
4881 0000D90B E8BCFEFFFF          <1>     call    sysvideo_15_12 ; window preparations
4882 0000D910 72F8                <1>     jc      short pix_op_cpy_1
4883                              <1>     ; ecx = bytes per row (to be applied)
4884                              <1>     ; edx = screen width in bytes
4885                              <1>     ; ebx = row count
4886 0000D912 8B3D[629D0100]      <1>     mov     edi, [v_str]
4887 0000D918 F605[589D0100]20    <1>     test    byte [v_ops], 20h ; masked copy ?
4888 0000D91F 7405                <1>     jz      short pix_op_cpy_w_0 ; no
4889 0000D921 E909100000          <1>     jmp     m_pix_op_cpy_w ; window copy except mask color
4890                              <1> pix_op_cpy_w_0:
4891                              <1>     ; esi = current row (virtual) addr in user's buff
4892                              <1>     ; edi = window's current row address (video mem)
4893                              <1>     ; ecx = transfer count per row
4894 0000D926 E809320000          <1>     call    transfer_from_user_buffer
4895 0000D92B 72DD                <1>     jc      short pix_op_cpy_1
4896 0000D92D 010D[348E0100]      <1>     add     [u.r0], ecx
4897 0000D933 4B                  <1>     dec     ebx
4898 0000D934 74D4                <1>     jz      short pix_op_cpy_1 ; ok.
4899                              <1>     ; next row
4900 0000D936 01CE                <1>     add     esi, ecx ; next row in user's buffer
4901 0000D938 01D7                <1>     add     edi, edx ; next row of window (system)
4902 0000D93A EBEA                <1>     jmp     short pix_op_cpy_w_0
4903                              <1>
4904                              <1> pix_op_add:
4905                              <1>     ; 31/01/2021
4906                              <1>     ; 30/01/2021
4907                              <1>     ; ADD COLOR
4908                              <1>     ;
4909                              <1>     ; INPUT:
4910                              <1>     ;   CL = color (8 bit, 256 colors)
4911                              <1>     ;  ECX = color (16 bit and true colors)
4912                              <1>     ;  EDX = start position (row, column)
4913                              <1>     ;         (HW = row, DX = column)
4914                              <1>     ;  ESI = size (rows, colums)
4915                              <1>     ;         (HW = rows, SI = columns)
4916                              <1>     ;
4917                              <1>     ;  [maskcolor] = mask color (to be excluded)
4918                              <1>     ;
4919                              <1>     ; OUTPUT:
4920                              <1>     ;       [u.r0] will be > 0 if succesful
4921                              <1>
4922 0000D93C F605[589D0100]10    <1>     test    byte [v_ops], 10h ; display page or window ?
```

```
4923 0000D943 7555                <1>     jnz     short pix_op_add_w ; window
4924                              <1>
4925 0000D945 8B3D[5A9D0100]      <1>     mov     edi, [v_mem]
4926 0000D94B 89FE                <1>     mov     esi, edi
4927                              <1>     ; ecx = color (CL, CX, ECX)
4928 0000D94D 89C8                <1>     mov     eax, ecx
4929 0000D94F 8B0D[5E9D0100]      <1>     mov     ecx, [v_siz] ; display page pixel count
4930                              <1>
4931 0000D955 F605[589D0100]20    <1>     test    byte [v_ops], 20h ; masked color adding ?
4932 0000D95C 7405                <1>     jz      short pix_op_add_0 ; no
4933 0000D95E E9CB100000          <1>     jmp     m_pix_op_add ; add color except mask color
4934                              <1> pix_op_add_0:
4935 0000D963 803D[599D0100]08    <1>     cmp     byte [v_bpp], 8 ; 8bpp
4936 0000D96A 7707                <1>     ja      short pix_op_add_1
4937                              <1>
4938                              <1>     ; 256 colors (8bpp)
4939 0000D96C E84C0A0000          <1>     call    pix_op_add_8
4940 0000D971 EB1E                <1>     jmp     short pix_op_add_4
4941                              <1>
4942                              <1> pix_op_add_1:
4943 0000D973 803D[599D0100]18    <1>     cmp     byte [v_bpp], 24 ; 24bpp
4944 0000D97A 7710                <1>     ja      short pix_op_add_3 ; 32bpp
4945 0000D97C 7207                <1>     jb      short pix_op_add_2 ; 16bpp
4946                              <1>
4947                              <1>     ; 24 bit true colors
4948 0000D97E E85A0A0000          <1>     call    pix_op_add_24
4949 0000D983 EB0C                <1>     jmp     short pix_op_add_4
4950                              <1>
4951                              <1>     ; 65536 colors (16bpp)
4952                              <1> pix_op_add_2:
4953 0000D985 E8410A0000          <1>     call    pix_op_add_16
4954 0000D98A EB05                <1>     jmp     short pix_op_add_4
4955                              <1>
4956                              <1>     ; 32 bit true colors
4957                              <1> pix_op_add_3:
4958 0000D98C E86C0A0000          <1>     call    pix_op_add_32
4959                              <1> pix_op_add_4:
4960 0000D991 29F7                <1>     sub     edi, esi
4961 0000D993 893D[348E0100]      <1>     mov     [u.r0], edi
4962                              <1> pix_op_add_5:
4963 0000D999 C3                  <1>     retn
4964                              <1>
4965                              <1> pix_op_add_w:
4966                              <1>     ; 31/01/2021
4967 0000D99A 51                  <1>     push    ecx ; * ; color
4968 0000D99B 89D1                <1>     mov     ecx, edx ; win start pos
4969 0000D99D 89F2                <1>     mov     edx, esi ; size (rows, cols)
4970 0000D99F E828FEFFFF          <1>     call    sysvideo_15_12 ; window preparations
4971 0000D9A4 58                  <1>     pop     eax ; * ; color
4972 0000D9A5 72F2                <1>     jc      short pix_op_add_5
4973                              <1>
4974 0000D9A7 F605[589D0100]20    <1>     test    byte [v_ops], 20h ; masked color adding ?
4975 0000D9AE 7405                <1>     jz      short pix_op_add_w_0 ; no
4976 0000D9B0 E927110000          <1>     jmp     m_pix_op_add_w
4977                              <1>                     ; window add color except mask color
4978                              <1> pix_op_add_w_0:
4979                              <1>     ; ecx = bytes per row (to be applied)
4980                              <1>     ; edx = screen width in bytes
4981                              <1>     ; ebx = row count
4982                              <1>     ; eax = color
4983                              <1>
4984 0000D9B5 8B3D[629D0100]      <1>     mov     edi, [v_str]
4985 0000D9BB 803D[599D0100]08    <1>     cmp     byte [v_bpp], 8 ; 8bpp
4986 0000D9C2 7707                <1>     ja      short pix_op_add_w_1
4987                              <1>
4988                              <1>     ; 256 colors (8bpp)
4989 0000D9C4 BD[BDE30000]        <1>     mov     ebp, pix_op_add_8
4990 0000D9C9 EB1E                <1>     jmp     short pix_op_add_w_4
4991                              <1>
4992                              <1> pix_op_add_w_1:
4993 0000D9CB 803D[599D0100]18    <1>     cmp     byte [v_bpp], 24 ; 24bpp
4994 0000D9D2 7710                <1>     ja      short pix_op_add_w_3 ; 32bpp
4995 0000D9D4 7207                <1>     jb      short pix_op_add_w_2 ; 16bpp
4996                              <1>
4997                              <1>     ; 24 bit true colors
4998 0000D9D6 BD[DDE30000]        <1>     mov     ebp, pix_op_add_24
4999 0000D9DB EB0C                <1>     jmp     short pix_op_add_w_4
5000                              <1>
5001                              <1>     ; 65536 colors (16bpp)
5002                              <1> pix_op_add_w_2:
5003 0000D9DD BD[CBE30000]        <1>     mov     ebp, pix_op_add_16
5004 0000D9E2 EB05                <1>     jmp     short pix_op_add_w_4
5005                              <1>
5006                              <1>     ; 32 bit true colors
5007                              <1> pix_op_add_w_3:
5008 0000D9E4 BD[FDE30000]        <1>     mov     ebp, pix_op_add_32
5009                              <1> pix_op_add_w_4:
5010 0000D9E9 E95F010000          <1>     jmp     pix_op_add_w_x
5011                              <1>
5012                              <1> pix_op_sub:
5013                              <1>     ; 31/01/2021
5014                              <1>     ; SUB COLOR
5015                              <1>     ;
5016                              <1>     ; INPUT:
5017                              <1>     ;   CL = color (8 bit, 256 colors)
5018                              <1>     ;  ECX = color (16 bit and true colors)
5019                              <1>     ;  EDX = start position (row, column)
5020                              <1>     ;        (HW = row, DX = column)
5021                              <1>     ;  ESI = size (rows, colums)
5022                              <1>     ;        (HW = rows, SI = columns)
5023                              <1>     ;
5024                              <1>     ;  [maskcolor] = mask color (to be excluded)
5025                              <1>     ;
5026                              <1>     ; OUTPUT:
5027                              <1>     ;       [u.r0] will be > 0 if succesful
5028                              <1>
5029 0000D9EE F605[589D0100]10    <1>     test    byte [v_ops], 10h ; display page or window ?
5030 0000D9F5 7555                <1>     jnz     short pix_op_sub_w ; window
5031                              <1>
5032 0000D9F7 8B3D[5A9D0100]      <1>     mov     edi, [v_mem]
5033 0000D9FD 89FE                <1>     mov     esi, edi
5034                              <1>     ; ecx = color (CL, CX, ECX)
5035 0000D9FF 89C8                <1>     mov     eax, ecx
5036 0000DA01 8B0D[5E9D0100]      <1>     mov     ecx, [v_siz] ; display page pixel count
5037                              <1>
5038 0000DA07 F605[589D0100]20    <1>     test    byte [v_ops], 20h ; masked color subtract ?
5039 0000DA0E 7405                <1>     jz      short pix_op_sub_0 ; no
5040 0000DA10 E9FA100000          <1>     jmp     m_pix_op_sub ; sub color except mask color
5041                              <1> pix_op_sub_0:
5042 0000DA15 803D[599D0100]08    <1>     cmp     byte [v_bpp], 8 ; 8bpp
5043 0000DA1C 7707                <1>     ja      short pix_op_sub_1
5044                              <1>
5045                              <1>     ; 256 colors (8bpp)
5046 0000DA1E E8E9090000          <1>     call    pix_op_sub_8
```

```
5047 0000DA23 EB1E                <1>     jmp     short pix_op_sub_4
5048                              <1>
5049                              <1> pix_op_sub_1:
5050 0000DA25 803D[599D0100]18    <1>     cmp     byte [v_bpp], 24 ; 24bpp
5051 0000DA2C 7710                <1>     ja      short pix_op_sub_3 ; 32bpp
5052 0000DA2E 7207                <1>     jb      short pix_op_sub_2 ; 16bpp
5053                              <1>
5054                              <1>     ; 24 bit true colors
5055 0000DA30 E8FA090000          <1>     call    pix_op_sub_24
5056 0000DA35 EB0C                <1>     jmp     short pix_op_sub_4
5057                              <1>
5058                              <1>     ; 65536 colors (16bpp)
5059                              <1> pix_op_sub_2:
5060 0000DA37 E8E0090000          <1>     call    pix_op_sub_16
5061 0000DA3C EB05                <1>     jmp     short pix_op_sub_4
5062                              <1>
5063                              <1>     ; 32 bit true colors
5064                              <1> pix_op_sub_3:
5065 0000DA3E E8060A0000          <1>     call    pix_op_sub_32
5066                              <1> pix_op_sub_4:
5067 0000DA43 29F7                <1>     sub     edi, esi
5068 0000DA45 893D[348E0100]      <1>     mov     [u.r0], edi
5069                              <1> pix_op_sub_5:
5070 0000DA4B C3                  <1>     retn
5071                              <1>
5072                              <1> pix_op_sub_w:
5073                              <1>     ; 31/01/2021
5074 0000DA4C 51                  <1>     push    ecx ; * ; color
5075 0000DA4D 89D1                <1>     mov     ecx, edx ; win start pos
5076 0000DA4F 89F2                <1>     mov     edx, esi ; size (rows, cols)
5077 0000DA51 E876FDFFFF          <1>     call    sysvideo_15_12 ; window preparations
5078 0000DA56 58                  <1>     pop     eax ; * ; color
5079 0000DA57 72F2                <1>     jc      short pix_op_sub_5
5080                              <1>
5081 0000DA59 F605[589D0100]20    <1>     test    byte [v_ops], 20h ; masked color subtract ?
5082 0000DA60 7405                <1>     jz      short pix_op_sub_w_0 ; no
5083 0000DA62 E94B110000          <1>     jmp     m_pix_op_sub_w
5084                              <1>                     ; window sub color except mask color
5085                              <1> pix_op_sub_w_0:
5086                              <1>     ; ecx = bytes per row (to be applied)
5087                              <1>     ; edx = screen width in bytes
5088                              <1>     ; ebx = row count
5089                              <1>     ; eax = color
5090                              <1>
5091 0000DA67 8B3D[629D0100]      <1>     mov     edi, [v_str]
5092 0000DA6D 803D[599D0100]08    <1>     cmp     byte [v_bpp], 8 ; 8bpp
5093 0000DA74 7707                <1>     ja      short pix_op_sub_w_1
5094                              <1>
5095                              <1>     ; 256 colors (8bpp)
5096 0000DA76 BD[0CE40000]        <1>     mov     ebp, pix_op_sub_8
5097 0000DA7B EB1E                <1>     jmp     short pix_op_sub_w_4
5098                              <1>
5099                              <1> pix_op_sub_w_1:
5100 0000DA7D 803D[599D0100]18    <1>     cmp     byte [v_bpp], 24 ; 24bpp
5101 0000DA84 7710                <1>     ja      short pix_op_sub_w_3 ; 32bpp
5102 0000DA86 7207                <1>     jb      short pix_op_sub_w_2 ; 16bpp
5103                              <1>
5104                              <1>     ; 24 bit true colors
5105 0000DA88 BD[2FE40000]        <1>     mov     ebp, pix_op_sub_24
5106 0000DA8D EB0C                <1>     jmp     short pix_op_sub_w_4
5107                              <1>
5108                              <1>     ; 65536 colors (16bpp)
5109                              <1> pix_op_sub_w_2:
5110 0000DA8F BD[1CE40000]        <1>     mov     ebp, pix_op_sub_16
5111 0000DA94 EB05                <1>     jmp     short pix_op_sub_w_4
5112                              <1>
5113                              <1>     ; 32 bit true colors
5114                              <1> pix_op_sub_w_3:
5115 0000DA96 BD[49E40000]        <1>     mov     ebp, pix_op_sub_32
5116                              <1> pix_op_sub_w_4:
5117 0000DA9B E9AD000000          <1>     jmp     pix_op_sub_w_x
5118                              <1>
5119                              <1> pix_op_mix:
5120                              <1>     ; 31/01/2021
5121                              <1>     ; MIX COLOR
5122                              <1>     ;
5123                              <1>     ; INPUT:
5124                              <1>     ;   CL = color (8 bit, 256 colors)
5125                              <1>     ;  ECX = color (16 bit and true colors)
5126                              <1>     ;  EDX = start position (row, column)
5127                              <1>     ;        (HW = row, DX = column)
5128                              <1>     ;  ESI = size (rows, colums)
5129                              <1>     ;        (HW = rows, SI = columns)
5130                              <1>     ;
5131                              <1>     ;  [maskcolor] = mask color (to be excluded)
5132                              <1>     ;
5133                              <1>     ; OUTPUT:
5134                              <1>     ;       [u.r0] will be > 0 if succesful
5135                              <1>
5136 0000DAA0 F605[589D0100]10    <1>     test    byte [v_ops], 10h ; display page or window ?
5137 0000DAA7 7555                <1>     jnz     short pix_op_mix_w ; window
5138                              <1>
5139 0000DAA9 8B3D[5A9D0100]      <1>     mov     edi, [v_mem]
5140 0000DAAF 89FE                <1>     mov     esi, edi
5141                              <1>     ; ecx = color (CL, CX, ECX)
5142 0000DAB1 89C8                <1>     mov     eax, ecx
5143 0000DAB3 8B0D[5E9D0100]      <1>     mov     ecx, [v_siz] ; display page pixel count
5144                              <1>
5145 0000DAB9 F605[589D0100]20    <1>     test    byte [v_ops], 20h ; masked color mix ?
5146 0000DAC0 7405                <1>     jz      short pix_op_mix_0 ; no
5147 0000DAC2 E91E110000          <1>     jmp     m_pix_op_mix ; mix colors except mask color
5148                              <1> pix_op_mix_0:
5149 0000DAC7 803D[599D0100]08    <1>     cmp     byte [v_bpp], 8 ; 8bpp
5150 0000DACE 7707                <1>     ja      short pix_op_mix_1
5151                              <1>
5152                              <1>     ; 256 colors (8bpp)
5153 0000DAD0 E8F3090000          <1>     call    pix_op_mix_8
5154 0000DAD5 EB1E                <1>     jmp     short pix_op_mix_4
5155                              <1>
5156                              <1> pix_op_mix_1:
5157 0000DAD7 803D[599D0100]18    <1>     cmp     byte [v_bpp], 24 ; 24bpp
5158 0000DADE 7710                <1>     ja      short pix_op_mix_3 ; 32bpp
5159 0000DAE0 7207                <1>     jb      short pix_op_mix_2 ; 16bpp
5160                              <1>
5161                              <1>     ; 24 bit true colors
5162 0000DAE2 E8FC090000          <1>     call    pix_op_mix_24
5163 0000DAE7 EB0C                <1>     jmp     short pix_op_mix_4
5164                              <1>
5165                              <1>     ; 65536 colors (16bpp)
5166                              <1> pix_op_mix_2:
5167 0000DAE9 E8E6090000          <1>     call    pix_op_mix_16
5168 0000DAEE EB05                <1>     jmp     short pix_op_mix_4
5169                              <1>
5170                              <1>     ; 32 bit true colors
```

```
                                <1> pix_op_mix_3:
0000DAF0 E80A0A0000              <1>     call    pix_op_mix_32
                                <1> pix_op_mix_4:
0000DAF5 29F7                    <1>     sub     edi, esi
0000DAF7 893D[348E0100]          <1>     mov     [u.r0], edi
                                <1> pix_op_mix_5:
0000DAFD C3                      <1>     retn
                                <1>
                                <1> pix_op_mix_w:
                                <1>     ; 31/01/2021
0000DAFE 51                      <1>     push    ecx ; * ; color
0000DAFF 89D1                    <1>     mov     ecx, edx ; win start pos
0000DB01 89F2                    <1>     mov     edx, esi ; size (rows, cols)
0000DB03 E8C4FCFFFF              <1>     call    sysvideo_15_12 ; window preparations
0000DB08 58                      <1>     pop     eax ; * ; color
0000DB09 72F2                    <1>     jc      short pix_op_mix_5
                                <1>
0000DB0B F605[589D0100]20        <1>     test    byte [v_ops], 20h ; masked color mix ?
0000DB12 7405                    <1>     jz      short pix_op_mix_w_0 ; no
0000DB14 E969110000              <1>     jmp     m_pix_op_mix_w
                                <1>                     ; window mix colors except mask color
                                <1> pix_op_mix_w_0:
                                <1>     ; ecx = bytes per row (to be applied)
                                <1>     ; edx = screen width in bytes
                                <1>     ; ebx = row count
                                <1>     ; eax = color
                                <1>
0000DB19 8B3D[629D0100]          <1>     mov     edi, [v_str]
0000DB1F 803D[599D0100]08        <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000DB26 7707                    <1>     ja      short pix_op_mix_w_1
                                <1>
                                <1>     ; 256 colors (8bpp)
0000DB28 BD[C8E40000]            <1>     mov     ebp, pix_op_mix_8
0000DB2D EB1E                    <1>     jmp     short pix_op_mix_w_x
                                <1>
                                <1> pix_op_mix_w_1:
0000DB2F 803D[599D0100]18        <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000DB36 7710                    <1>     ja      short pix_op_mix_w_3 ; 32bpp
0000DB38 7207                    <1>     jb      short pix_op_mix_w_2 ; 16bpp
                                <1>
                                <1>     ; 24 bit true colors
0000DB3A BD[E3E40000]            <1>     mov     ebp, pix_op_mix_24
0000DB3F EB0C                    <1>     jmp     short pix_op_mix_w_x
                                <1>
                                <1>     ; 65536 colors (16bpp)
                                <1> pix_op_mix_w_2:
0000DB41 BD[D4E40000]            <1>     mov     ebp, pix_op_mix_16
0000DB46 EB05                    <1>     jmp     short pix_op_mix_w_x
                                <1>
                                <1>     ; 32 bit true colors
                                <1> pix_op_mix_w_3:
0000DB48 BD[FFE40000]            <1>     mov     ebp, pix_op_mix_32
                                <1>     ;jmp    short pix_op_mix_w_x
                                <1>
                                <1> pix_op_mix_w_x:
                                <1> pix_op_add_w_x:
                                <1> pix_op_sub_w_x:
                                <1> pix_op_rpl_w_x:
                                <1> pix_op_orc_w_x:
                                <1> pix_op_and_w_x:
                                <1> pix_op_xor_w_x:
                                <1>     ; 27/02/2021
                                <1>     ; 31/01/2021
                                <1>     ; ecx = bytes per row (to be applied)
                                <1>     ; edx = windows (screen) width in bytes
                                <1>     ; ebx = row count
                                <1>     ; eax = color
                                <1>     ; ebp = pixel operation subroutine address
0000DB4D 52                      <1>     push    edx
0000DB4E 51                      <1>     push    ecx
0000DB4F 57                      <1>     push    edi
0000DB50 8B0D[6E9D0100]          <1>     mov     ecx, [pixcount] ; 27/02/2021
0000DB56 FFD5                    <1>     call    ebp ; call pixel-row operation
0000DB58 5F                      <1>     pop     edi
0000DB59 59                      <1>     pop     ecx ; bytes per row
0000DB5A 010D[348E0100]          <1>     add     [u.r0], ecx
0000DB60 5A                      <1>     pop     edx
0000DB61 01D7                    <1>     add     edi, edx ; next row
0000DB63 4B                      <1>     dec     ebx
0000DB64 75E7                    <1>     jnz     short pix_op_mix_w_x
0000DB66 C3                      <1>     retn
                                <1>
                                <1> pix_op_rpl:
                                <1>     ; 01/02/2021
                                <1>     ; REPLACE COLOR
                                <1>     ;
                                <1>     ; INPUT:
                                <1>     ;   CL = old/current color (8 bit, 256 colors)
                                <1>     ;  ECX = old/current color (16 bit and true colors)
                                <1>     ;   DL = new color (8 bit, 256 colors)
                                <1>     ;  EDX = new color (16 bit and true colors)
                                <1>     ;  ESI = start position (row, column)
                                <1>     ;        (HW = row, DX = column)
                                <1>     ;  EDI = size (rows, colums)
                                <1>     ;        (HW = rows, SI = columns)
                                <1>     ; OUTPUT:
                                <1>     ;       [u.r0] will be > 0 if succesful
                                <1>
0000DB67 F605[589D0100]10        <1>     test    byte [v_ops], 10h ; display page or window ?
0000DB6E 754D                    <1>     jnz     short pix_op_rpl_w ; window
                                <1>
0000DB70 8B3D[5A9D0100]          <1>     mov     edi, [v_mem]
0000DB76 89FE                    <1>     mov     esi, edi
                                <1>     ; ecx = old color (CL, CX, ECX) -to be replaced with-
                                <1>     ; edx = new color (CL, CX, ECX) -new one-
0000DB78 89D0                    <1>     mov     eax, edx ; new color
0000DB7A 890D[6A9D0100]          <1>     mov     [maskcolor], ecx ; old color
0000DB80 8B0D[5E9D0100]          <1>     mov     ecx, [v_siz] ; display page pixel count
                                <1> pix_op_rpl_0:
0000DB86 803D[599D0100]08        <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000DB8D 7707                    <1>     ja      short pix_op_rpl_1
                                <1>
                                <1>     ; 256 colors (8bpp)
0000DB8F E82F0A0000              <1>     call    pix_op_rpl_8
0000DB94 EB1E                    <1>     jmp     short pix_op_rpl_4
                                <1>
                                <1> pix_op_rpl_1:
0000DB96 803D[599D0100]18        <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000DB9D 7710                    <1>     ja      short pix_op_rpl_3 ; 32bpp
0000DB9F 7207                    <1>     jb      short pix_op_rpl_2 ; 16bpp
                                <1>
                                <1>     ; 24 bit true colors
0000DBA1 E8400A0000              <1>     call    pix_op_rpl_24
0000DBA6 EB0C                    <1>     jmp     short pix_op_rpl_4
```

```
                                <1>
                                <1>     ; 65536 colors (16bpp)
                                <1> pix_op_rpl_2:
0000DBA8 E8260A0000             <1>     call    pix_op_rpl_16
0000DBAD EB05                   <1>     jmp     short pix_op_rpl_4
                                <1>
                                <1>     ; 32 bit true colors
                                <1> pix_op_rpl_3:
0000DBAF E8540A0000             <1>     call    pix_op_rpl_32
                                <1> pix_op_rpl_4:
0000DBB4 29F7                   <1>     sub     edi, esi
0000DBB6 893D[348E0100]         <1>     mov     [u.r0], edi
                                <1> pix_op_rpl_5:
0000DBBC C3                     <1>     retn
                                <1>
                                <1> pix_op_rpl_w:
                                <1>     ; 01/02/2021
0000DBBD 890D[6A9D0100]         <1>     mov     [maskcolor], ecx ; old color
0000DBC3 52                     <1>     push    edx ; * ; new color
0000DBC4 89F1                   <1>     mov     ecx, esi ; win start pos
0000DBC6 89FA                   <1>     mov     edx, edi ; size (rows, cols)
0000DBC8 E8FFFBFFFF             <1>     call    sysvideo_15_12 ; window preparations
0000DBCD 58                     <1>     pop     eax ; * ; new color
0000DBCE 72EC                   <1>     jc      short pix_op_rpl_5
                                <1>
                                <1>     ; replace window color
                                <1> pix_op_rpl_w_0:
                                <1>     ; ecx = bytes per row (to be applied)
                                <1>     ; edx = screen width in bytes
                                <1>     ; ebx = row count
                                <1>     ; eax = new color
                                <1>     ; [maskcolor] = old color
                                <1>
0000DBD0 8B3D[629D0100]         <1>     mov     edi, [v_str]
                                <1>
0000DBD6 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000DBDD 7707                   <1>     ja      short pix_op_rpl_w_1
                                <1>
                                <1>     ; 256 colors (8bpp)
0000DBDF BD[C3E50000]           <1>     mov     ebp, pix_op_rpl_8
0000DBE4 EB1E                   <1>     jmp     short pix_op_rpl_w_4
                                <1>
                                <1> pix_op_rpl_w_1:
0000DBE6 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000DBED 7710                   <1>     ja      short pix_op_rpl_w_3 ; 32bpp
0000DBEF 7207                   <1>     jb      short pix_op_rpl_w_2 ; 16bpp
                                <1>
                                <1>     ; 24 bit true colors
0000DBF1 BD[E6E50000]           <1>     mov     ebp, pix_op_rpl_24
0000DBF6 EB0C                   <1>     jmp     short pix_op_rpl_w_4
                                <1>
                                <1>     ; 65536 colors (16bpp)
                                <1> pix_op_rpl_w_2:
0000DBF8 BD[D3E50000]           <1>     mov     ebp, pix_op_rpl_16
0000DBFD EB05                   <1>     jmp     short pix_op_rpl_w_4
                                <1>
                                <1>     ; 32 bit true colors
                                <1> pix_op_rpl_w_3:
0000DBFF BD[08E60000]           <1>     mov     ebp, pix_op_rpl_32
                                <1> pix_op_rpl_w_4:
0000DC04 E944FFFFFF             <1>     jmp     pix_op_rpl_w_x
                                <1>
                                <1> pix_op_orc:
                                <1>     ; 31/01/2021
                                <1>     ; OR COLOR
                                <1>     ;
                                <1>     ; INPUT:
                                <1>     ;   CL = color (8 bit, 256 colors)
                                <1>     ;  ECX = color (16 bit and true colors)
                                <1>     ;  EDX = start position (row, column)
                                <1>     ;        (HW = row, DX = column)
                                <1>     ;  ESI = size (rows, colums)
                                <1>     ;        (HW = rows, SI = columns)
                                <1>     ;
                                <1>     ;  [maskcolor] = mask color (to be excluded)
                                <1>     ;
                                <1>     ; OUTPUT:
                                <1>     ;       [u.r0] will be > 0 if succesful
                                <1>
0000DC09 F605[589D0100]10       <1>     test    byte [v_ops], 10h ; display page or window ?
0000DC10 7555                   <1>     jnz     short pix_op_or_w ; window
                                <1>
0000DC12 8B3D[5A9D0100]         <1>     mov     edi, [v_mem]
0000DC18 89FE                   <1>     mov     esi, edi
                                <1>     ; ecx = color (CL, CX, ECX)
0000DC1A 89C8                   <1>     mov     eax, ecx
0000DC1C 8B0D[5E9D0100]         <1>     mov     ecx, [v_siz] ; display page pixel count
                                <1>
0000DC22 F605[589D0100]20       <1>     test    byte [v_ops], 20h ; masked color 'or' ?
0000DC29 7405                   <1>     jz      short pix_op_or_0 ; no
0000DC2B E945110000             <1>     jmp     m_pix_op_or ; 'or' color except mask color
                                <1> pix_op_or_0:
0000DC30 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000DC37 7707                   <1>     ja      short pix_op_or_1
                                <1>
                                <1>     ; 256 colors (8bpp)
0000DC39 E81B080000             <1>     call    pix_op_or_8
0000DC3E EB1E                   <1>     jmp     short pix_op_or_4
                                <1>
                                <1> pix_op_or_1:
0000DC40 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000DC47 7710                   <1>     ja      short pix_op_or_3 ; 32bpp
0000DC49 7207                   <1>     jb      short pix_op_or_2 ; 16bpp
                                <1>
                                <1>     ; 24 bit true colors
0000DC4B E817080000             <1>     call    pix_op_or_24
0000DC50 EB0C                   <1>     jmp     short pix_op_or_4
                                <1>
                                <1>     ; 65536 colors (16bpp)
                                <1> pix_op_or_2:
0000DC52 E808080000             <1>     call    pix_op_or_16
0000DC57 EB05                   <1>     jmp     short pix_op_or_4
                                <1>
                                <1>     ; 32 bit true colors
                                <1> pix_op_or_3:
0000DC59 E818080000             <1>     call    pix_op_or_32
                                <1> pix_op_or_4:
0000DC5E 29F7                   <1>     sub     edi, esi
0000DC60 893D[348E0100]         <1>     mov     [u.r0], edi
                                <1> pix_op_or_5:
0000DC66 C3                     <1>     retn
                                <1>
                                <1> pix_op_or_w:
                                <1>     ; 31/01/2021
```

```
5419 0000DC67 51                <1>     push    ecx ; * ; color
5420 0000DC68 89D1              <1>     mov     ecx, edx ; win start pos
5421 0000DC6A 89F2              <1>     mov     edx, esi ; size (rows, cols)
5422 0000DC6C E85BFBFFFF        <1>     call    sysvideo_15_12 ; window preparations
5423 0000DC71 58                <1>     pop     eax ; * ; color
5424 0000DC72 72F2              <1>     jc      short pix_op_or_5
5425                            <1>
5426 0000DC74 F605[589D0100]20  <1>     test    byte [v_ops], 20h ; masked color 'or' ?
5427 0000DC7B 7405              <1>     jz      short pix_op_or_w_0 ; no
5428 0000DC7D E980110000        <1>     jmp     m_pix_op_or_w
5429                            <1>                     ; window 'or' color except mask color
5430                            <1> pix_op_or_w_0:
5431                            <1>     ; ecx = bytes per row (to be applied)
5432                            <1>     ; edx = screen width in bytes
5433                            <1>     ; ebx = row count
5434                            <1>     ; eax = color
5435                            <1>
5436 0000DC82 8B3D[629D0100]    <1>     mov     edi, [v_str]
5437 0000DC88 803D[599D0100]08  <1>     cmp     byte [v_bpp], 8 ; 8bpp
5438 0000DC8F 7707              <1>     ja      short pix_op_or_w_1
5439                            <1>
5440                            <1>     ; 256 colors (8bpp)
5441 0000DC91 BD[59E40000]      <1>     mov     ebp, pix_op_or_8
5442 0000DC96 EB1E              <1>     jmp     short pix_op_or_w_4
5443                            <1>
5444                            <1> pix_op_or_w_1:
5445 0000DC98 803D[599D0100]18  <1>     cmp     byte [v_bpp], 24 ; 24bpp
5446 0000DC9F 7710              <1>     ja      short pix_op_or_w_3 ; 32bpp
5447 0000DCA1 7207              <1>     jb      short pix_op_or_w_2 ; 16bpp
5448                            <1>
5449                            <1>     ; 24 bit true colors
5450 0000DCA3 BD[67E40000]      <1>     mov     ebp, pix_op_or_24
5451 0000DCA8 EB0C              <1>     jmp     short pix_op_or_w_4
5452                            <1>
5453                            <1>     ; 65536 colors (16bpp)
5454                            <1> pix_op_or_w_2:
5455 0000DCAA BD[5FE40000]      <1>     mov     ebp, pix_op_or_16
5456 0000DCAF EB05              <1>     jmp     short pix_op_or_w_4
5457                            <1>
5458                            <1>     ; 32 bit true colors
5459                            <1> pix_op_or_w_3:
5460 0000DCB1 BD[76E40000]      <1>     mov     ebp, pix_op_or_32
5461                            <1> pix_op_or_w_4:
5462 0000DCB6 E992FEFFFF        <1>     jmp     pix_op_orc_w_x
5463                            <1>
5464                            <1> pix_op_and:
5465                            <1>     ; 31/01/2021
5466                            <1>     ; AND COLOR
5467                            <1>     ;
5468                            <1>     ; INPUT:
5469                            <1>     ;   CL = color (8 bit, 256 colors)
5470                            <1>     ;  ECX = color (16 bit and true colors)
5471                            <1>     ;  EDX = start position (row, column)
5472                            <1>     ;        (HW = row, DX = column)
5473                            <1>     ;  ESI = size (rows, colums)
5474                            <1>     ;        (HW = rows, SI = columns)
5475                            <1>     ;
5476                            <1>     ;  [maskcolor] = mask color (to be excluded)
5477                            <1>     ;
5478                            <1>     ; OUTPUT:
5479                            <1>     ;       [u.r0] will be > 0 if succesful
5480                            <1>
5481 0000DCBB F605[589D0100]10  <1>     test    byte [v_ops], 10h ; display page or window ?
5482 0000DCC2 7555              <1>     jnz     short pix_op_and_w ; window
5483                            <1>
5484 0000DCC4 8B3D[5A9D0100]    <1>     mov     edi, [v_mem]
5485 0000DCCA 89FE              <1>     mov     esi, edi
5486                            <1>     ; ecx = color (CL, CX, ECX)
5487 0000DCCC 89C8              <1>     mov     eax, ecx
5488 0000DCCE 8B0D[5E9D0100]    <1>     mov     ecx, [v_siz] ; display page pixel count
5489                            <1>
5490 0000DCD4 F605[589D0100]20  <1>     test    byte [v_ops], 20h ; masked color 'and' ?
5491 0000DCDB 7405              <1>     jz      short pix_op_and_0 ; no
5492 0000DCDD E9D30F0000        <1>     jmp     m_pix_op_and ; 'and' color except mask color
5493                            <1> pix_op_and_0:
5494 0000DCE2 803D[599D0100]08  <1>     cmp     byte [v_bpp], 8 ; 8bpp
5495 0000DCE9 7707              <1>     ja      short pix_op_and_1
5496                            <1>
5497                            <1>     ; 256 colors (8bpp)
5498 0000DCEB E88E070000        <1>     call    pix_op_and_8
5499 0000DCF0 EB1E              <1>     jmp     short pix_op_and_4
5500                            <1>
5501                            <1> pix_op_and_1:
5502 0000DCF2 803D[599D0100]18  <1>     cmp     byte [v_bpp], 24 ; 24bpp
5503 0000DCF9 7710              <1>     ja      short pix_op_and_3 ; 32bpp
5504 0000DCFB 7207              <1>     jb      short pix_op_and_2 ; 16bpp
5505                            <1>
5506                            <1>     ; 24 bit true colors
5507 0000DCFD E88A070000        <1>     call    pix_op_and_24
5508 0000DD02 EB0C              <1>     jmp     short pix_op_and_4
5509                            <1>
5510                            <1>     ; 65536 colors (16bpp)
5511                            <1> pix_op_and_2:
5512 0000DD04 E87B070000        <1>     call    pix_op_and_16
5513 0000DD09 EB05              <1>     jmp     short pix_op_and_4
5514                            <1>
5515                            <1>     ; 32 bit true colors
5516                            <1> pix_op_and_3:
5517 0000DD0B E88B070000        <1>     call    pix_op_and_32
5518                            <1> pix_op_and_4:
5519 0000DD10 29F7              <1>     sub     edi, esi
5520 0000DD12 893D[348E0100]    <1>     mov     [u.r0], edi
5521                            <1> pix_op_and_5:
5522 0000DD18 C3                <1>     retn
5523                            <1>
5524                            <1> pix_op_and_w:
5525                            <1>     ; 31/01/2021
5526 0000DD19 51                <1>     push    ecx ; * ; color
5527 0000DD1A 89D1              <1>     mov     ecx, edx ; win start pos
5528 0000DD1C 89F2              <1>     mov     edx, esi ; size (rows, cols)
5529 0000DD1E E8A9FAFFFF        <1>     call    sysvideo_15_12 ; window preparations
5530 0000DD23 58                <1>     pop     eax ; * ; color
5531 0000DD24 72F2              <1>     jc      short pix_op_and_5
5532                            <1>
5533 0000DD26 F605[589D0100]20  <1>     test    byte [v_ops], 20h ; masked color 'and' ?
5534 0000DD2D 7405              <1>     jz      short pix_op_and_w_0 ; no
5535 0000DD2F E90E100000        <1>     jmp     m_pix_op_and_w
5536                            <1>                     ; window 'and' color except mask color
5537                            <1> pix_op_and_w_0:
5538                            <1>     ; ecx = bytes per row (to be applied)
5539                            <1>     ; edx = screen width in bytes
5540                            <1>     ; ebx = row count
5541                            <1>     ; eax = color
5542                            <1>
```

```
5543 0000DD34 8B3D[629D0100]      <1>     mov     edi, [v_str]
5544 0000DD3A 803D[599D0100]08    <1>     cmp     byte [v_bpp], 8 ; 8bpp
5545 0000DD41 7707                <1>     ja      short pix_op_and_w_1
5546                              <1>
5547                              <1>     ; 256 colors (8bpp)
5548 0000DD43 BD[7EE40000]        <1>     mov     ebp, pix_op_and_8
5549 0000DD48 EB1E                <1>     jmp     short pix_op_and_w_4
5550                              <1>
5551                              <1> pix_op_and_w_1:
5552 0000DD4A 803D[599D0100]18    <1>     cmp     byte [v_bpp], 24 ; 24bpp
5553 0000DD51 7710                <1>     ja      short pix_op_and_w_3 ; 32bpp
5554 0000DD53 7207                <1>     jb      short pix_op_and_w_2 ; 16bpp
5555                              <1>
5556                              <1>     ; 24 bit true colors
5557 0000DD55 BD[8CE40000]        <1>     mov     ebp, pix_op_and_24
5558 0000DD5A EB0C                <1>     jmp     short pix_op_and_w_4
5559                              <1>
5560                              <1>     ; 65536 colors (16bpp)
5561                              <1> pix_op_and_w_2:
5562 0000DD5C BD[84E40000]        <1>     mov     ebp, pix_op_and_16
5563 0000DD61 EB05                <1>     jmp     short pix_op_and_w_4
5564                              <1>
5565                              <1>     ; 32 bit true colors
5566                              <1> pix_op_and_w_3:
5567 0000DD63 BD[9BE40000]        <1>     mov     ebp, pix_op_and_32
5568                              <1> pix_op_and_w_4:
5569 0000DD68 E9E0FDFFFF          <1>     jmp     pix_op_and_w_x
5570                              <1>
5571                              <1> pix_op_xor:
5572                              <1>     ; 31/01/2021
5573                              <1>     ; XOR COLOR
5574                              <1>     ;
5575                              <1>     ; INPUT:
5576                              <1>     ;   CL = color (8 bit, 256 colors)
5577                              <1>     ;  ECX = color (16 bit and true colors)
5578                              <1>     ;  EDX = start position (row, column)
5579                              <1>     ;        (HW = row, DX = column)
5580                              <1>     ;  ESI = size (rows, colums)
5581                              <1>     ;        (HW = rows, SI = columns)
5582                              <1>     ;
5583                              <1>     ;  [maskcolor] = mask color (to be excluded)
5584                              <1>     ;
5585                              <1>     ; OUTPUT:
5586                              <1>     ;       [u.r0] will be > 0 if succesful
5587                              <1>
5588 0000DD6D F605[589D0100]10    <1>     test    byte [v_ops], 10h ; display page or window ?
5589 0000DD74 7555                <1>     jnz     short pix_op_xor_w ; window
5590                              <1>
5591 0000DD76 8B3D[5A9D0100]      <1>     mov     edi, [v_mem]
5592 0000DD7C 89FE                <1>     mov     esi, edi
5593                              <1>     ; ecx = color (CL, CX, ECX)
5594 0000DD7E 89C8                <1>     mov     eax, ecx
5595 0000DD80 8B0D[5E9D0100]      <1>     mov     ecx, [v_siz] ; display page pixel count
5596                              <1>
5597 0000DD86 F605[589D0100]20    <1>     test    byte [v_ops], 20h ; masked color 'xor' ?
5598 0000DD8D 7405                <1>     jz      short pix_op_xor_0 ; no
5599 0000DD8F E9A1100000          <1>     jmp     m_pix_op_xor ; 'xor' color except mask color
5600                              <1> pix_op_xor_0:
5601 0000DD94 803D[599D0100]08    <1>     cmp     byte [v_bpp], 8 ; 8bpp
5602 0000DD9B 7707                <1>     ja      short pix_op_xor_1
5603                              <1>
5604                              <1>     ; 256 colors (8bpp)
5605 0000DD9D E801070000          <1>     call    pix_op_xor_8
5606 0000DDA2 EB1E                <1>     jmp     short pix_op_xor_4
5607                              <1>
5608                              <1> pix_op_xor_1:
5609 0000DDA4 803D[599D0100]18    <1>     cmp     byte [v_bpp], 24 ; 24bpp
5610 0000DDAB 7710                <1>     ja      short pix_op_xor_3 ; 32bpp
5611 0000DDAD 7207                <1>     jb      short pix_op_xor_2 ; 16bpp
5612                              <1>
5613                              <1>     ; 24 bit true colors
5614 0000DDAF E8FD060000          <1>     call    pix_op_xor_24
5615 0000DDB4 EB0C                <1>     jmp     short pix_op_xor_4
5616                              <1>
5617                              <1>     ; 65536 colors (16bpp)
5618                              <1> pix_op_xor_2:
5619 0000DDB6 E8EE060000          <1>     call    pix_op_xor_16
5620 0000DDBB EB05                <1>     jmp     short pix_op_xor_4
5621                              <1>
5622                              <1>     ; 32 bit true colors
5623                              <1> pix_op_xor_3:
5624 0000DDBD E8FE060000          <1>     call    pix_op_xor_32
5625                              <1> pix_op_xor_4:
5626 0000DDC2 29F7                <1>     sub     edi, esi
5627 0000DDC4 893D[348E0100]      <1>     mov     [u.r0], edi
5628                              <1> pix_op_xor_5:
5629 0000DDCA C3                  <1>     retn
5630                              <1>
5631                              <1> pix_op_xor_w:
5632                              <1>     ; 31/01/2021
5633 0000DDCB 51                  <1>     push    ecx ; * ; color
5634 0000DDCC 89D1                <1>     mov     ecx, edx ; win start pos
5635 0000DDCE 89F2                <1>     mov     edx, esi ; size (rows, cols)
5636 0000DDD0 E8F7F9FFFF          <1>     call    sysvideo_15_12 ; window preparations
5637 0000DDD5 58                  <1>     pop     eax ; * ; color
5638 0000DDD6 72F2                <1>     jc      short pix_op_xor_5
5639                              <1>
5640 0000DDD8 F605[589D0100]20    <1>     test    byte [v_ops], 20h ; masked color 'xor' ?
5641 0000DDDF 7405                <1>     jz      short pix_op_xor_w_0 ; no
5642 0000DDE1 E9DC100000          <1>     jmp     m_pix_op_xor_w
5643                              <1>                     ; window 'xor' color except mask color
5644                              <1> pix_op_xor_w_0:
5645                              <1>     ; ecx = bytes per row (to be applied)
5646                              <1>     ; edx = screen width in bytes
5647                              <1>     ; ebx = row count
5648                              <1>     ; eax = color
5649                              <1>
5650 0000DDE6 8B3D[629D0100]      <1>     mov     edi, [v_str]
5651 0000DDEC 803D[599D0100]08    <1>     cmp     byte [v_bpp], 8 ; 8bpp
5652 0000DDF3 7707                <1>     ja      short pix_op_xor_w_1
5653                              <1>
5654                              <1>     ; 256 colors (8bpp)
5655 0000DDF5 BD[A3E40000]        <1>     mov     ebp, pix_op_xor_8
5656 0000DDFA EB1E                <1>     jmp     short pix_op_xor_w_4
5657                              <1>
5658                              <1> pix_op_xor_w_1:
5659 0000DDFC 803D[599D0100]18    <1>     cmp     byte [v_bpp], 24 ; 24bpp
5660 0000DE03 7710                <1>     ja      short pix_op_xor_w_3 ; 32bpp
5661 0000DE05 7207                <1>     jb      short pix_op_xor_w_2 ; 16bpp
5662                              <1>
5663                              <1>     ; 24 bit true colors
5664 0000DE07 BD[B1E40000]        <1>     mov     ebp, pix_op_xor_24
5665 0000DE0C EB0C                <1>     jmp     short pix_op_xor_w_4
5666                              <1>
```

```
                                <1>     ; 65536 colors (16bpp)
                                <1> pix_op_xor_w_2:
0000DE0E BD[A9E40000]           <1>     mov     ebp, pix_op_xor_16
0000DE13 EB05                   <1>     jmp     short pix_op_xor_w_4
                                <1>
                                <1>     ; 32 bit true colors
                                <1> pix_op_xor_w_3:
0000DE15 BD[C0E40000]           <1>     mov     ebp, pix_op_xor_32
                                <1> pix_op_xor_w_4:
0000DE1A E92EFDFFFF             <1>     jmp     pix_op_xor_w_x
                                <1>
                                <1> pix_op_new:
                                <1>     ; 31/01/2021
                                <1>     ; 30/01/2021
                                <1>     ; CHANGE COLOR
                                <1>     ;
                                <1>     ; INPUT:
                                <1>     ;   CL = color (8 bit, 256 colors)
                                <1>     ;  ECX = color (16 bit and true colors)
                                <1>     ;  EDX = start position (row, column)
                                <1>     ;        (HW = row, DX = column)
                                <1>     ;  ESI = size (rows, colums)
                                <1>     ;        (HW = rows, SI = columns)
                                <1>     ;
                                <1>     ;  [maskcolor] = mask color (to be excluded)
                                <1>     ;
                                <1>     ; OUTPUT:
                                <1>     ;       [u.r0] will be > 0 if succesful
                                <1>
0000DE1F F605[589D0100]10       <1>     test    byte [v_ops], 10h ; display page or window ?
0000DE26 7554                   <1>     jnz     short pix_op_new_w ; window
                                <1>
0000DE28 8B3D[5A9D0100]         <1>     mov     edi, [v_mem]
0000DE2E 89FE                   <1>     mov     esi, edi
                                <1>     ; ecx = color (CL, CX, ECX)
0000DE30 89C8                   <1>     mov     eax, ecx
0000DE32 8B0D[5E9D0100]         <1>     mov     ecx, [v_siz] ; display page pixel count
                                <1>
0000DE38 F605[589D0100]20       <1>     test    byte [v_ops], 20h ; masked color change ?
0000DE3F 7405                   <1>     jz      short pix_op_new_0 ; no
0000DE41 E90A0B0000             <1>     jmp     m_pix_op_new ; change color except mask color
                                <1> pix_op_new_0:
0000DE46 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000DE4D 7706                   <1>     ja      short pix_op_new_2
                                <1>
                                <1>     ; 256 colors (8bpp)
                                <1> pix_op_new_1:
0000DE4F 88C4                   <1>     mov     ah, al
0000DE51 D1E9                   <1>     shr     ecx, 1
0000DE53 EB12                   <1>     jmp     short pix_op_new_3
                                <1>
                                <1> pix_op_new_2:
0000DE55 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000DE5C 7713                   <1>     ja      short pix_op_new_4 ; 32bpp
0000DE5E 7207                   <1>     jb      short pix_op_new_3 ; 16bpp
                                <1>
                                <1>     ; 31/01/2021
                                <1>
                                <1>     ; 24 bit true colors
0000DE60 E849050000             <1>     call    pix_op_new_24
                                <1>
0000DE65 EB0C                   <1>     jmp     short pix_op_new_5
                                <1>
                                <1>     ; 65536 colors (16bpp)
                                <1> pix_op_new_3:
0000DE67 89C2                   <1>     mov     edx, eax
0000DE69 C1E010                 <1>     shl     eax, 16
0000DE6C 6689D0                 <1>     mov     ax, dx
0000DE6F D1E9                   <1>     shr     ecx, 1 ; dword counts
                                <1>     ; 32 bit true colors
                                <1> pix_op_new_4:
0000DE71 F3AB                   <1>     rep     stosd
                                <1> pix_op_new_5:
0000DE73 29F7                   <1>     sub     edi, esi
0000DE75 893D[348E0100]         <1>     mov     [u.r0], edi
                                <1> pix_op_new_6:
0000DE7B C3                     <1>     retn
                                <1>
                                <1> pix_op_new_w:
                                <1>     ; 31/01/2021
                                <1>     ; 30/01/2021
0000DE7C 51                     <1>     push    ecx ; * ; color
0000DE7D 89D1                   <1>     mov     ecx, edx ; win start pos
0000DE7F 89F2                   <1>     mov     edx, esi ; size (rows, cols)
0000DE81 E846F9FFFF             <1>     call    sysvideo_15_12 ; window preparations
0000DE86 58                     <1>     pop     eax ; * ; color
0000DE87 72F2                   <1>     jc      short pix_op_new_6
                                <1>
0000DE89 F605[589D0100]20       <1>     test    byte [v_ops], 20h ; masked color change ?
0000DE90 7405                   <1>     jz      short pix_op_new_w_0 ; no
0000DE92 E9470B0000             <1>     jmp     m_pix_op_new_w
                                <1>                     ; window chg color except mask color
                                <1> pix_op_new_w_0:
                                <1>     ; ecx = bytes per row (to be applied)
                                <1>     ; edx = screen width in bytes
                                <1>     ; ebx = row count
                                <1>     ; eax = color
                                <1>
0000DE97 8B3D[629D0100]         <1>     mov     edi, [v_str]
                                <1>
0000DE9D 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000DEA4 7707                   <1>     ja      short pix_op_new_w_1
                                <1>
                                <1>     ; 256 colors (8bpp)
0000DEA6 BD[A7E30000]           <1>     mov     ebp, pix_op_new_8
0000DEAB EB1E                   <1>     jmp     short pix_op_new_w_x
                                <1>
                                <1> pix_op_new_w_1:
0000DEAD 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000DEB4 7710                   <1>     ja      short pix_op_new_w_3 ; 32bpp
0000DEB6 7207                   <1>     jb      short pix_op_new_w_2 ; 16bpp
                                <1>
                                <1>     ; 24 bit true colors
0000DEB8 BD[AEE30000]           <1>     mov     ebp, pix_op_new_24
0000DEBD EB0C                   <1>     jmp     short pix_op_new_w_x
                                <1>
                                <1>     ; 65536 colors (16bpp)
                                <1> pix_op_new_w_2:
0000DEBF BD[AAE30000]           <1>     mov     ebp, pix_op_new_16
0000DEC4 EB05                   <1>     jmp     short pix_op_new_w_x
                                <1>
                                <1>     ; 32 bit true colors
                                <1> pix_op_new_w_3:
0000DEC6 BD[BAE30000]           <1>     mov     ebp, pix_op_new_32
```

```
                                <1>     ;jmp    short pix_op_new_w_x
                                <1>
                                <1> pix_op_new_w_x:
                                <1> pix_op_not_w_x:
                                <1> pix_op_neg_w_x:
                                <1> pix_op_inc_w_x:
                                <1> pix_op_dec_w_x:
                                <1>     ; 27/02/2021
                                <1>     ; 01/02/2021
                                <1>     ; 31/01/2021
                                <1>     ; ecx = bytes per row (to be applied)
                                <1>     ; edx = windows (screen) width in bytes
                                <1>     ; ebx = row count
                                <1>     ; eax = color
                                <1>     ; ebp = pixel operation subroutine address
                                <1>     ;push   edx ; 01/02/2021
0000DECB 51                     <1>     push    ecx
0000DECC 57                     <1>     push    edi
0000DECD 8B0D[6E9D0100]         <1>     mov     ecx, [pixcount] ; 27/02/2021
0000DED3 FFD5                   <1>     call    ebp ; call pixel-row operation
0000DED5 5F                     <1>     pop     edi
0000DED6 59                     <1>     pop     ecx ; bytes per row
0000DED7 010D[348E0100]         <1>     add     [u.r0], ecx
                                <1>     ;pop    edx ; 01/02/2021
0000DEDD 01D7                   <1>     add     edi, edx ; next row
0000DEDF 4B                     <1>     dec     ebx
0000DEE0 75E9                   <1>     jnz     short pix_op_new_w_x
0000DEE2 C3                     <1>     retn
                                <1>
                                <1> pix_op_not:
                                <1>     ; 31/01/2021
                                <1>     ; NOT COLOR
                                <1>     ;
                                <1>     ; INPUT:
                                <1>     ;  ECX = start position (row, column)
                                <1>     ;        (HW = row, CX = column)
                                <1>     ;  EDX = size (rows, colums)
                                <1>     ;        (HW = rows, DX = columns)
                                <1>     ;         (0 -> invalid
                                <1>     ;          (1 -> horizontal or vertical line)
                                <1>     ;  [maskcolor] = mask color (to be excluded)
                                <1>     ;
                                <1>     ; OUTPUT:
                                <1>     ;       [u.r0] will be > 0 if succesful
                                <1>
0000DEE3 F605[589D0100]10       <1>     test    byte [v_ops], 10h ; display page or window ?
0000DEEA 7553                   <1>     jnz     short pix_op_not_w ; window
                                <1>
0000DEEC 8B3D[5A9D0100]         <1>     mov     edi, [v_mem]
0000DEF2 89FE                   <1>     mov     esi, edi
0000DEF4 8B0D[5E9D0100]         <1>     mov     ecx, [v_siz] ; display page pixel count
                                <1>
0000DEFA F605[589D0100]20       <1>     test    byte [v_ops], 20h ; masked color 'not' ?
0000DF01 7405                   <1>     jz      short pix_op_not_0 ; no
0000DF03 E9ED0F0000             <1>     jmp     m_pix_op_not ; 'not' color except mask color
                                <1> pix_op_not_0:
0000DF08 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000DF0F 7707                   <1>     ja      short pix_op_not_1
                                <1>
                                <1>     ; 256 colors (8bpp)
0000DF11 E8F5050000             <1>     call    pix_op_not_8
0000DF16 EB1E                   <1>     jmp     short pix_op_not_4
                                <1>
                                <1> pix_op_not_1:
0000DF18 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000DF1F 7710                   <1>     ja      short pix_op_not_3 ; 32bpp
0000DF21 7207                   <1>     jb      short pix_op_not_2 ; 16bpp
                                <1>
                                <1>     ; 24 bit true colors
0000DF23 E8F1050000             <1>     call    pix_op_not_24
0000DF28 EB0C                   <1>     jmp     short pix_op_not_4
                                <1>
                                <1>     ; 65536 colors (16bpp)
                                <1> pix_op_not_2:
0000DF2A E8E2050000             <1>     call    pix_op_not_16
0000DF2F EB05                   <1>     jmp     short pix_op_not_4
                                <1>
                                <1>     ; 32 bit true colors
                                <1> pix_op_not_3:
0000DF31 E8EE050000             <1>     call    pix_op_not_32
                                <1> pix_op_not_4:
0000DF36 29F7                   <1>     sub     edi, esi
0000DF38 893D[348E0100]         <1>     mov     [u.r0], edi
                                <1> pix_op_not_5:
0000DF3E C3                     <1>     retn
                                <1>
                                <1> pix_op_not_w:
                                <1>     ; 31/01/2021
                                <1>     ; ecx = win start pos (row, column)
                                <1>     ; edx = size (rows, columns)
0000DF3F E888F8FFFF             <1>     call    sysvideo_15_12 ; window preparations
0000DF44 72F8                   <1>     jc      short pix_op_not_5
                                <1>
0000DF46 F605[589D0100]20       <1>     test    byte [v_ops], 20h ; masked color 'not' ?
0000DF4D 7405                   <1>     jz      short pix_op_not_w_0 ; no
0000DF4F E926100000             <1>     jmp     m_pix_op_not_w
                                <1>                     ; window 'not' color except mask color
                                <1> pix_op_not_w_0:
                                <1>     ; ecx = bytes per row (to be applied)
                                <1>     ; edx = screen width in bytes
                                <1>     ; ebx = row count
                                <1>
0000DF54 8B3D[629D0100]         <1>     mov     edi, [v_str]
                                <1>
0000DF5A 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000DF61 7707                   <1>     ja      short pix_op_not_w_1
                                <1>
                                <1>     ; 256 colors (8bpp)
0000DF63 BD[0BE50000]           <1>     mov     ebp, pix_op_not_8
0000DF68 EB1E                   <1>     jmp     short pix_op_not_w_4
                                <1>
                                <1> pix_op_not_w_1:
0000DF6A 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000DF71 7710                   <1>     ja      short pix_op_not_w_3 ; 32bpp
0000DF73 7207                   <1>     jb      short pix_op_not_w_2 ; 16bpp
                                <1>
                                <1>     ; 24 bit true colors
0000DF75 BD[19E50000]           <1>     mov     ebp, pix_op_not_24
0000DF7A EB0C                   <1>     jmp     short pix_op_not_w_4
                                <1>
                                <1>     ; 65536 colors (16bpp)
                                <1> pix_op_not_w_2:
0000DF7C BD[11E50000]           <1>     mov     ebp, pix_op_not_16
0000DF81 EB05                   <1>     jmp     short pix_op_not_w_4
```

```
                                <1>
                                <1>     ; 32 bit true colors
                                <1> pix_op_not_w_3:
0000DF83 BD[24E50000]           <1>     mov     ebp, pix_op_not_32
                                <1> pix_op_not_w_4:
0000DF88 E93EFFFFFF             <1>     jmp     pix_op_not_w_x
                                <1>
                                <1> pix_op_neg:
                                <1>     ; 31/01/2021
                                <1>     ; NEGATE COLOR
                                <1>     ;
                                <1>     ; INPUT:
                                <1>     ;  ECX = start position (row, column)
                                <1>     ;         (HW = row, CX = column)
                                <1>     ;  EDX = size (rows, colums)
                                <1>     ;         (HW = rows, DX = columns)
                                <1>     ;        (0 -> invalid
                                <1>     ;         (1 -> horizontal or vertical line)
                                <1>     ;  [maskcolor] = mask color (to be excluded)
                                <1>     ;
                                <1>     ; OUTPUT:
                                <1>     ;       [u.r0] will be > 0 if succesful
                                <1>
0000DF8D F605[589D0100]10       <1>     test    byte [v_ops], 10h ; display page or window ?
0000DF94 7553                   <1>     jnz     short pix_op_neg_w ; window
                                <1>
0000DF96 8B3D[5A9D0100]         <1>     mov     edi, [v_mem]
0000DF9C 89FE                   <1>     mov     esi, edi
0000DF9E 8B0D[5E9D0100]         <1>     mov     ecx, [v_siz] ; display page pixel count
                                <1>
0000DFA4 F605[589D0100]20       <1>     test    byte [v_ops], 20h ; masked negate color ?
0000DFAB 7405                   <1>     jz      short pix_op_neg_0 ; no
0000DFAD E9FB0F0000             <1>     jmp     m_pix_op_neg ; 'neg' color except mask color
                                <1> pix_op_neg_0:
0000DFB2 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000DFB9 7707                   <1>     ja      short pix_op_neg_1
                                <1>
                                <1>     ; 256 colors (8bpp)
0000DFBB E86C050000             <1>     call    pix_op_neg_8
0000DFC0 EB1E                   <1>     jmp     short pix_op_neg_4
                                <1>
                                <1> pix_op_neg_1:
0000DFC2 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000DFC9 7710                   <1>     ja      short pix_op_neg_3 ; 32bpp
0000DFCB 7207                   <1>     jb      short pix_op_neg_2 ; 16bpp
                                <1>
                                <1>     ; 24 bit true colors
0000DFCD E868050000             <1>     call    pix_op_neg_24
0000DFD2 EB0C                   <1>     jmp     short pix_op_neg_4
                                <1>
                                <1>     ; 65536 colors (16bpp)
                                <1> pix_op_neg_2:
0000DFD4 E859050000             <1>     call    pix_op_neg_16
0000DFD9 EB05                   <1>     jmp     short pix_op_neg_4
                                <1>
                                <1>     ; 32 bit true colors
                                <1> pix_op_neg_3:
0000DFDB E86C050000             <1>     call    pix_op_neg_32
                                <1> pix_op_neg_4:
0000DFE0 29F7                   <1>     sub     edi, esi
0000DFE2 893D[348E0100]         <1>     mov     [u.r0], edi
                                <1> pix_op_neg_5:
0000DFE8 C3                     <1>     retn
                                <1>
                                <1> pix_op_neg_w:
                                <1>     ; 31/01/2021
                                <1>     ; ecx = win start pos (row, column)
                                <1>     ; edx = size (rows, columns)
0000DFE9 E8DEF7FFFF             <1>     call    sysvideo_15_12 ; window preparations
0000DFEE 72F8                   <1>     jc      short pix_op_neg_5
                                <1>
0000DFF0 F605[589D0100]20       <1>     test    byte [v_ops], 20h ; masked negate color ?
0000DFF7 7405                   <1>     jz      short pix_op_neg_w_0 ; no
0000DFF9 E934100000             <1>     jmp     m_pix_op_neg_w
                                <1>                     ; window 'neg' color except mask color
                                <1> pix_op_neg_w_0:
                                <1>     ; ecx = bytes per row (to be applied)
                                <1>     ; edx = screen width in bytes
                                <1>     ; ebx = row count
                                <1>
0000DFFE 8B3D[629D0100]         <1>     mov     edi, [v_str]
                                <1>
0000E004 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000E00B 7707                   <1>     ja      short pix_op_neg_w_1
                                <1>
                                <1>     ; 256 colors (8bpp)
0000E00D BD[2CE50000]           <1>     mov     ebp, pix_op_neg_8
0000E012 EB1E                   <1>     jmp     short pix_op_neg_w_4
                                <1>
                                <1> pix_op_neg_w_1:
0000E014 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000E01B 7710                   <1>     ja      short pix_op_neg_w_3 ; 32bpp
0000E01D 7207                   <1>     jb      short pix_op_neg_w_2 ; 16bpp
                                <1>
                                <1>     ; 24 bit true colors
0000E01F BD[3AE50000]           <1>     mov     ebp, pix_op_neg_24
0000E024 EB0C                   <1>     jmp     short pix_op_neg_w_4
                                <1>
                                <1>     ; 65536 colors (16bpp)
                                <1> pix_op_neg_w_2:
0000E026 BD[32E50000]           <1>     mov     ebp, pix_op_neg_16
0000E02B EB05                   <1>     jmp     short pix_op_neg_w_4
                                <1>
                                <1>     ; 32 bit true colors
                                <1> pix_op_neg_w_3:
0000E02D BD[4CE50000]           <1>     mov     ebp, pix_op_neg_32
                                <1> pix_op_neg_w_4:
0000E032 E994FEFFFF             <1>     jmp     pix_op_neg_w_x
                                <1>
                                <1> pix_op_inc:
                                <1>     ; 31/01/2021
                                <1>     ; INCREASE COLOR
                                <1>     ;
                                <1>     ; INPUT:
                                <1>     ;  ECX = start position (row, column)
                                <1>     ;         (HW = row, CX = column)
                                <1>     ;  EDX = size (rows, colums)
                                <1>     ;         (HW = rows, DX = columns)
                                <1>     ;        (0 -> invalid
                                <1>     ;         (1 -> horizontal or vertical line)
                                <1>     ;  [maskcolor] = mask color (to be excluded)
                                <1>     ;
                                <1>     ; OUTPUT:
                                <1>     ;       [u.r0] will be > 0 if succesful
```

```
                                <1>
0000E037 F605[589D0100]10       <1>     test    byte [v_ops], 10h ; display page or window ?
0000E03E 7553                   <1>     jnz     short pix_op_inc_w ; window
                                <1>
0000E040 8B3D[5A9D0100]         <1>     mov     edi, [v_mem]
0000E046 89FE                   <1>     mov     esi, edi
0000E048 8B0D[5E9D0100]         <1>     mov     ecx, [v_siz] ; display page pixel count
                                <1>
0000E04E F605[589D0100]20       <1>     test    byte [v_ops], 20h ; masked increase color ?
0000E055 7405                   <1>     jz      short pix_op_inc_0 ; no
0000E057 E909100000             <1>     jmp     m_pix_op_inc ; 'inc' color except mask color
                                <1> pix_op_inc_0:
0000E05C 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000E063 7707                   <1>     ja      short pix_op_inc_1
                                <1>
                                <1>     ; 256 colors (8bpp)
0000E065 E8EA040000             <1>     call    pix_op_inc_8
0000E06A EB1E                   <1>     jmp     short pix_op_inc_4
                                <1>
                                <1> pix_op_inc_1:
0000E06C 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000E073 7710                   <1>     ja      short pix_op_inc_3 ; 32bpp
0000E075 7207                   <1>     jb      short pix_op_inc_2 ; 16bpp
                                <1>
                                <1>     ; 24 bit true colors
0000E077 E8EF040000             <1>     call    pix_op_inc_24
0000E07C EB0C                   <1>     jmp     short pix_op_inc_4
                                <1>
                                <1>     ; 65536 colors (16bpp)
                                <1> pix_op_inc_2:
0000E07E E8DB040000             <1>     call    pix_op_inc_16
0000E083 EB05                   <1>     jmp     short pix_op_inc_4
                                <1>
                                <1>     ; 32 bit true colors
                                <1> pix_op_inc_3:
0000E085 E8F5040000             <1>     call    pix_op_inc_32
                                <1> pix_op_inc_4:
0000E08A 29F7                   <1>     sub     edi, esi
0000E08C 893D[348E0100]         <1>     mov     [u.r0], edi
                                <1> pix_op_inc_5:
0000E092 C3                     <1>     retn
                                <1>
                                <1> pix_op_inc_w:
                                <1>     ; 31/01/2021
                                <1>     ; ecx = win start pos (row, column)
                                <1>     ; edx = size (rows, columns)
0000E093 E834F7FFFF             <1>     call    sysvideo_15_12 ; window preparations
0000E098 72F8                   <1>     jc      short pix_op_inc_5
                                <1>
0000E09A F605[589D0100]20       <1>     test    byte [v_ops], 20h ; masked increase color ?
0000E0A1 7405                   <1>     jz      short pix_op_inc_w_0 ; no
0000E0A3 E956100000             <1>     jmp     m_pix_op_inc_w
                                <1>                     ; window 'inc' color except mask color
                                <1> pix_op_inc_w_0:
                                <1>     ; ecx = bytes per row (to be applied)
                                <1>     ; edx = screen width in bytes
                                <1>     ; ebx = row count
                                <1>
0000E0A8 8B3D[629D0100]         <1>     mov     edi, [v_str]
                                <1>
0000E0AE 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000E0B5 7707                   <1>     ja      short pix_op_inc_w_1
                                <1>
                                <1>     ; 256 colors (8bpp)
0000E0B7 BD[54E50000]           <1>     mov     ebp, pix_op_inc_8
0000E0BC EB1E                   <1>     jmp     short pix_op_inc_w_4
                                <1>
                                <1> pix_op_inc_w_1:
0000E0BE 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000E0C5 7710                   <1>     ja      short pix_op_inc_w_3 ; 32bpp
0000E0C7 7207                   <1>     jb      short pix_op_inc_w_2 ; 16bpp
                                <1>
                                <1>     ; 24 bit true colors
0000E0C9 BD[6BE50000]           <1>     mov     ebp, pix_op_inc_24
0000E0CE EB0C                   <1>     jmp     short pix_op_inc_w_4
                                <1>
                                <1>     ; 65536 colors (16bpp)
                                <1> pix_op_inc_w_2:
0000E0D0 BD[5EE50000]           <1>     mov     ebp, pix_op_inc_16
0000E0D5 EB05                   <1>     jmp     short pix_op_inc_w_4
                                <1>
                                <1>     ; 32 bit true colors
                                <1> pix_op_inc_w_3:
0000E0D7 BD[7FE50000]           <1>     mov     ebp, pix_op_inc_32
                                <1> pix_op_inc_w_4:
0000E0DC E9EAFDFFFF             <1>     jmp     pix_op_inc_w_x
                                <1>
                                <1> pix_op_dec:
                                <1>     ; 31/01/2021
                                <1>     ; DECREASE COLOR
                                <1>     ;
                                <1>     ; INPUT:
                                <1>     ;  ECX = start position (row, column)
                                <1>     ;         (HW = row, CX = column)
                                <1>     ;  EDX = size (rows, colums)
                                <1>     ;         (HW = rows, DX = columns)
                                <1>     ;        (0 -> invalid
                                <1>     ;         (1 -> horizontal or vertical line)
                                <1>     ;  [maskcolor] = mask color (to be excluded)
                                <1>     ;
                                <1>     ; OUTPUT:
                                <1>     ;       [u.r0] will be > 0 if succesful
                                <1>
0000E0E1 F605[589D0100]10       <1>     test    byte [v_ops], 10h ; display page or window ?
0000E0E8 7553                   <1>     jnz     short pix_op_dec_w ; window
                                <1>
0000E0EA 8B3D[5A9D0100]         <1>     mov     edi, [v_mem]
0000E0F0 89FE                   <1>     mov     esi, edi
0000E0F2 8B0D[5E9D0100]         <1>     mov     ecx, [v_siz] ; display page pixel count
                                <1>
0000E0F8 F605[589D0100]20       <1>     test    byte [v_ops], 20h ; masked decrease color ?
0000E0FF 7405                   <1>     jz      short pix_op_dec_0 ; no
0000E101 E92B100000             <1>     jmp     m_pix_op_dec ; 'dec' color except mask color
                                <1> pix_op_dec_0:
0000E106 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000E10D 7707                   <1>     ja      short pix_op_dec_1
                                <1>
                                <1>     ; 256 colors (8bpp)
0000E10F E877040000             <1>     call    pix_op_dec_8
0000E114 EB1E                   <1>     jmp     short pix_op_dec_4
                                <1>
                                <1> pix_op_dec_1:
0000E116 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000E11D 7710                   <1>     ja      short pix_op_dec_3 ; 32bpp
```

```
6163 0000E11F 7207               <1>     jb      short pix_op_dec_2 ; 16bpp
6164                             <1>
6165                             <1>     ; 24 bit true colors
6166 0000E121 E87C040000         <1>     call    pix_op_dec_24
6167 0000E126 EB0C               <1>     jmp     short pix_op_dec_4
6168                             <1>
6169                             <1>     ; 65536 colors (16bpp)
6170                             <1> pix_op_dec_2:
6171 0000E128 E868040000         <1>     call    pix_op_dec_16
6172 0000E12D EB05               <1>     jmp     short pix_op_dec_4
6173                             <1>
6174                             <1>     ; 32 bit true colors
6175                             <1> pix_op_dec_3:
6176 0000E12F E881040000         <1>     call    pix_op_dec_32
6177                             <1> pix_op_dec_4:
6178 0000E134 29F7               <1>     sub     edi, esi
6179 0000E136 893D[348E0100]     <1>     mov     [u.r0], edi
6180                             <1> pix_op_dec_5:
6181 0000E13C C3                 <1>     retn
6182                             <1>
6183                             <1> pix_op_dec_w:
6184                             <1>     ; 31/01/2021
6185                             <1>     ; ecx = win start pos (row, column)
6186                             <1>     ; edx = size (rows, columns)
6187 0000E13D E88AF6FFFF         <1>     call    sysvideo_15_12 ; window preparations
6188 0000E142 72F8               <1>     jc      short pix_op_dec_5
6189                             <1>
6190 0000E144 F605[589D0100]20   <1>     test    byte [v_ops], 20h ; masked decrease color ?
6191 0000E14B 7405               <1>     jz      short pix_op_dec_w_0 ; no
6192 0000E14D E973100000         <1>     jmp     m_pix_op_dec_w
6193                             <1>                     ; window 'dec' color except mask color
6194                             <1> pix_op_dec_w_0:
6195                             <1>     ; ecx = bytes per row (to be applied)
6196                             <1>     ; edx = screen width in bytes
6197                             <1>     ; ebx = row count
6198                             <1>
6199 0000E152 8B3D[629D0100]     <1>     mov     edi, [v_str]
6200                             <1>
6201 0000E158 803D[599D0100]08   <1>     cmp     byte [v_bpp], 8 ; 8bpp
6202 0000E15F 7707               <1>     ja      short pix_op_dec_w_1
6203                             <1>
6204                             <1>     ; 256 colors (8bpp)
6205 0000E161 BD[8BE50000]       <1>     mov     ebp, pix_op_dec_8
6206 0000E166 EB1E               <1>     jmp     short pix_op_dec_w_4
6207                             <1>
6208                             <1> pix_op_dec_w_1:
6209 0000E168 803D[599D0100]18   <1>     cmp     byte [v_bpp], 24 ; 24bpp
6210 0000E16F 7710               <1>     ja      short pix_op_dec_w_3 ; 32bpp
6211 0000E171 7207               <1>     jb      short pix_op_dec_w_2 ; 16bpp
6212                             <1>
6213                             <1>     ; 24 bit true colors
6214 0000E173 BD[A2E50000]       <1>     mov     ebp, pix_op_dec_24
6215 0000E178 EB0C               <1>     jmp     short pix_op_dec_w_4
6216                             <1>
6217                             <1>     ; 65536 colors (16bpp)
6218                             <1> pix_op_dec_w_2:
6219 0000E17A BD[95E50000]       <1>     mov     ebp, pix_op_dec_16
6220 0000E17F EB05               <1>     jmp     short pix_op_dec_w_4
6221                             <1>
6222                             <1>     ; 32 bit true colors
6223                             <1> pix_op_dec_w_3:
6224 0000E181 BD[B5E50000]       <1>     mov     ebp, pix_op_dec_32
6225                             <1> pix_op_dec_w_4:
6226 0000E186 E940FDFFFF         <1>     jmp     pix_op_dec_w_x
6227                             <1>
6228                             <1> pix_op_blk:
6229                             <1>     ; 11/08/2022 (TRDOS 386 Kernel v2.0.5)
6230                             <1>     ; 23/01/2021
6231                             <1>     ; 22/02/2021
6232                             <1>     ; 02/02/2021
6233                             <1>     ; COPY PIXEL BLOCK -system to system-
6234                             <1>     ; WRITE PIXEL BLOCKS -user to system-
6235                             <1>     ;
6236                             <1>     ; INPUT:
6237                             <1>     ;   -If BL bit 5 is 0-
6238                             <1>     ;    ECX = start position (row, column) (*)
6239                             <1>     ;    (HW = row, CX = column)
6240                             <1>     ;    EDX = size (rows, colums) (*)
6241                             <1>     ;    (HW = rows, DX = columns)
6242                             <1>     ;         (0 -> invalid)
6243                             <1>     ;         (1 -> horizontal or vertical line)
6244                             <1>     ;    ESI = destination (row, column) (***)
6245                             <1>     ;   -If BL bit 5 is 1-
6246                             <1>     ;     CL = color (8 bit, 256 colors)
6247                             <1>     ;    ECX = color (16 bit and true colors)
6248                             <1>     ;    EDX = count of blocks (not bytes)
6249                             <1>     ;       (limit: 2048 blocks/windows)
6250                             <1>     ;    ESI = user's buffer address
6251                             <1>     ;         contains 64 bit block data
6252                             <1>     ;         BLOCK ADDRESS - (row, col), dword
6253                             <1>     ;         (first 32 bits)
6254                             <1>     ;         BLOCK SIZE - (rows, cols), dword
6255                             <1>     ;         (second 32 bits)
6256                             <1>     ; OUTPUT:
6257                             <1>     ;       [u.r0] will be > 0 if succesful
6258                             <1>
6259                             <1>     ; Window option ([v_ops] bit 4) will be ignored
6260                             <1>     ; (Function is used for display page coordinates)
6261                             <1>
6262 0000E18B F605[589D0100]20   <1>     test    byte [v_ops], 20h ; masked or direct ?
6263 0000E192 755A               <1>     jnz     short pix_op_blk_u ; blocks from user's buffer
6264                             <1>
6265 0000E194 89F0               <1>     mov     eax, esi ; destination position (row, col)
6266 0000E196 E8DEF6FFFF         <1>     call    calc_pixel_offset
6267 0000E19B 3B05[5E9D0100]     <1>     cmp     eax, [v_siz]
6268 0000E1A1 734A               <1>     jnb     short pix_op_blk_retn ; out of display page
6269 0000E1A3 89C6               <1>     mov     esi, eax
6270 0000E1A5 E8ACF6FFFF         <1>     call    pixels_to_byte_count
6271 0000E1AA 89C7               <1>     mov     edi, eax
6272 0000E1AC 89D0               <1>     mov     eax, edx  ; size
6273 0000E1AE E8C6F6FFFF         <1>     call    calc_pixel_offset
6274                             <1>     ; 22/02/2021
6275 0000E1B3 3B05[5E9D0100]     <1>     cmp     eax, [v_siz]
6276 0000E1B9 7732               <1>     ja      short pix_op_blk_retn ; out of display page
6277 0000E1BB 01C6               <1>     add     esi, eax
6278 0000E1BD 3B35[5E9D0100]     <1>     cmp     esi, [v_siz]
6279 0000E1C3 7728               <1>     ja      short pix_op_blk_retn ; out of display page
6280                             <1>
6281 0000E1C5 033D[5A9D0100]     <1>     add     edi, [v_mem] ; destination address
6282                             <1>
6283                             <1>     ; 23/01/2021
6284                             <1>     ;call   pixels_to_byte_count
6285                             <1>     ;add    edi, eax
6286                             <1>     ;jc     short pix_op_blk_retn ; out of display page
```

```
                                <1>     ;cmp    edi, [v_end]
                                <1>     ;ja     short pix_op_blk_retn ; out of display page
                                <1>     ;sub    edi, eax
                                <1>
0000E1CB E8FCF5FFFF             <1>     call    sysvideo_15_12 ; window preparations
0000E1D0 721B                   <1>     jc      short pix_op_blk_retn ; something wrong !?
                                <1>     ; ecx = bytes per row (to be applied)
                                <1>     ; edx = screen width in bytes
                                <1>     ; ebx = row count
                                <1>
0000E1D2 8B35[629D0100]         <1>     mov     esi, [v_str] ; source address
                                <1>
                                <1>     ; Note:
                                <1>     ; ecx & edx are already adjusted for pixel sizes
                                <1>     ; so, following code is proper all pixel sizes
                                <1>
0000E1D8 29CA                   <1>     sub     edx, ecx ; screen width - window width
                                <1> pix_op_blk_0:
0000E1DA 89C8                   <1>     mov     eax, ecx
0000E1DC 0105[348E0100]         <1>     add     [u.r0], eax
0000E1E2 F3A4                   <1>     rep     movsb
0000E1E4 89C1                   <1>     mov     ecx, eax
0000E1E6 01D6                   <1>     add     esi, edx ; next row
0000E1E8 01D7                   <1>     add     edi, edx ; next row
0000E1EA 4B                     <1>     dec     ebx
0000E1EB 75ED                   <1>     jnz     short pix_op_blk_0
                                <1> pix_op_blk_retn:
0000E1ED C3                     <1>     retn
                                <1>
                                <1> pix_op_blk_u:
                                <1>     ; fill blocks (windows) with desired color
                                <1>     ; according to block definitions in user's buffer
0000E1EE 81FA00080000           <1>     cmp     edx, 2048
0000E1F4 7605                   <1>     jna     short pix_op_blk_u_0
                                <1>     ; Maximum 2048 blocks
0000E1F6 BA00080000             <1>     mov     edx, 2048
                                <1> pix_op_blk_u_0:
0000E1FB 8025[589D0100]DF       <1>     and     byte [v_ops], ~20h ; clear masked bit
0000E202 890D[6A9D0100]         <1>     mov     [maskcolor], ecx ; save pixel color
                                <1>     ; 22/02/2021
                                <1>     ;mov    ebp, edx ; save blocks count
                                <1>     ;push   ebp
                                <1> pix_op_blk_u_next:
0000E208 52                     <1>     push    edx
                                <1>     ;mov    ecx, 8
                                <1>     ; 11/08/2022
0000E209 29C9                   <1>     sub     ecx, ecx
0000E20B B108                   <1>     mov     cl, 8
0000E20D BF[729D0100]           <1>     mov     edi, buffer8 ; 8 bytes small buffer
                                <1>     ; esi = user's buffer address
0000E212 E81D290000             <1>     call    transfer_from_user_buffer
0000E217 72D4                   <1>     jc      short pix_op_blk_retn
0000E219 01CE                   <1>     add     esi, ecx ; 22/02/2021
0000E21B 56                     <1>     push    esi
0000E21C 8B15[729D0100]         <1>     mov     edx, [buffer8] ; block start pos (row,col)
0000E222 8B35[769D0100]         <1>     mov     esi, [buffer8+4] ; block size (rows,cols)
0000E228 8B0D[6A9D0100]         <1>     mov     ecx, [maskcolor]
0000E22E E849FCFFFF             <1>     call    pix_op_new_w ; new (change) color (window)
0000E233 5E                     <1>     pop     esi
                                <1>     ;pop    ebp
                                <1>     ;dec    ebp
0000E234 5A                     <1>     pop     edx
0000E235 4A                     <1>     dec     edx
0000E236 75D0                   <1>     jnz     short pix_op_blk_u_next
0000E238 C3                     <1>     retn
                                <1>
                                <1> pix_op_lin:
                                <1>     ; 11/08/2022
                                <1>     ; 23/07/2022 - TRDOS 386 v2.0.5
                                <1>     ; 12/02/2021
                                <1>     ; 11/02/2021
                                <1>     ; 10/02/2021
                                <1>     ; 05/02/2021
                                <1>     ; 02/02/2021
                                <1>     ; WRITE LINE -direct-
                                <1>     ; WRITE LINE(S) -via user's buffer-
                                <1>     ;
                                <1>     ; INPUT:
                                <1>     ;   -If BL bit 5 is 0-
                                <1>     ;     CL = color (8 bit, 256 colors)
                                <1>     ;    ECX = color (16 bit and true colors)
                                <1>     ;     DX = low 12 bits - size (length)
                                <1>     ;          high 4 bits - direction or type
                                <1>     ;            0 - Horizontal line
                                <1>     ;            1 - Vertical line
                                <1>     ;          > 1 - undefined, invalid
                                <1>     ;    ESI = start position (row, column)
                                <1>     ;          (HW = row, SI = column)
                                <1>     ;   -If BL bit 5 is 1-
                                <1>     ;     CL = color (8 bit, 256 colors)
                                <1>     ;    ECX = color (16 bit and true colors)
                                <1>     ;     DX = number of lines (in user buffer)
                                <1>     ;          (limit: 2048 lines)
                                <1>     ;    ESI = user's buffer
                                <1>     ;          contains 64 bit data for lines
                                <1>     ;          START POINT: 32 bit (row, col)
                                <1>     ;          LENGTH: 32 bit
                                <1>     ;                  high 16 bits - 0
                                <1>     ;                  bit 0-11 - length
                                <1>     ;                  bit 12-15 - type (length)
                                <1>     ; OUTPUT:
                                <1>     ;       [u.r0] will be > 0 if succesful
                                <1>
                                <1>     ; Window option ([v_ops] bit 4) will be ignored
                                <1>     ; (Function is used for display page coordinates)
                                <1>
                                <1>     ; 10/02/2021
0000E239 F605[589D0100]20       <1>     test    byte [v_ops], 20h ; masked or direct ?
0000E240 7444                   <1>     jz      short pix_op_lin_vh ; direct (v/h lines)
                                <1>
                                <1>     ; lines from user's buffer
                                <1> pix_op_lin_u:
                                <1>     ; draw lines with desired color
                                <1>     ; according to line definitions in user's buffer
0000E242 81FA00080000           <1>     cmp     edx, 2048
0000E248 7605                   <1>     jna     short pix_op_lin_u_0
                                <1>     ; Maximum 2048 lines
0000E24A BA00080000             <1>     mov     edx, 2048
                                <1> pix_op_lin_u_0:
0000E24F 890D[6A9D0100]         <1>     mov     [maskcolor], ecx ; save pixel color
0000E255 89D5                   <1>     mov     ebp, edx ; save line count
                                <1> pix_op_lin_u_next:
                                <1>     ;mov    ecx, 8
                                <1>     ; 11/08/2022
```

```
6411 0000E257 29C9                <1>     sub     ecx, ecx
6412 0000E259 B108                <1>     mov     cl, 8
6413 0000E25B BF[729D0100]        <1>     mov     edi, buffer8 ; 8 bytes small buffer
6414                              <1>     ; esi = user's buffer address
6415 0000E260 E8CF280000          <1>     call    transfer_from_user_buffer
6416 0000E265 721E                <1>     jc      short pix_op_lin_retn
6417 0000E267 01CE                <1>     add     esi, ecx ; 11/02/2021
6418 0000E269 56                  <1>     push    esi
6419 0000E26A 8B35[729D0100]      <1>     mov     esi, [buffer8] ; line start pos (row,col)
6420 0000E270 8B15[769D0100]      <1>     mov     edx, [buffer8+4] ; line length
6421 0000E276 8B0D[6A9D0100]      <1>     mov     ecx, [maskcolor]
6422 0000E27C E805000000          <1>     call    pix_op_lin_vh ; new (change) color (window)
6423 0000E281 5E                  <1>     pop     esi
6424 0000E282 4D                  <1>     dec     ebp
6425 0000E283 75D2                <1>     jnz     short pix_op_lin_u_next
6426                              <1> pix_op_lin_retn:
6427 0000E285 C3                  <1>     retn
6428                              <1>
6429                              <1> pix_op_lin_vh:
6430 0000E286 81FA38140000        <1>     cmp     edx, 1438h ; 1920*1080 (780hx438h) limit
6431 0000E28C 7762                <1>     ja      short pix_op_lin_err1 ; invalid type
6432                              <1>                           ; (for current version)
6433 0000E28E 66F7C2FF0F          <1>     test    dx, 0FFFh
6434 0000E293 745B                <1>     jz      short pix_op_lin_err1 ; zero length!
6435                              <1>
6436 0000E295 89F0                <1>     mov     eax, esi ; start point (row, col)
6437 0000E297 E8DDF5FFFF          <1>     call    calc_pixel_offset
6438 0000E29C 3B05[5E9D0100]      <1>     cmp     eax, [v_siz]
6439 0000E2A2 734C                <1>     jnb     short pix_op_lin_err1 ; out of display page!
6440 0000E2A4 E8ADF5FFFF          <1>     call    pixels_to_byte_count
6441 0000E2A9 89C7                <1>     mov     edi, eax ; start point offset
6442 0000E2AB 033D[5A9D0100]      <1>     add     edi, [v_mem] ; LFB start address
6443 0000E2B1 89C8                <1>     mov     eax, ecx ; color
6444                              <1>
6445 0000E2B3 F6C610              <1>     test    dh, 10h
6446                              <1>     ;jz     pix_op_lin_h ; Horizontal line
6447                              <1>     ; 23/07/2022
6448 0000E2B6 7505                <1>     jnz     short pix_op_lin_v
6449 0000E2B8 E98A000000          <1>     jmp     pix_op_lin_h
6450                              <1>
6451                              <1> pix_op_lin_v:
6452                              <1>     ; Vertical line
6453 0000E2BD 80E60F              <1>     and     dh, 0Fh ; low 12 bits
6454 0000E2C0 51                  <1>     push    ecx ; color
6455 0000E2C1 89D1                <1>     mov     ecx, edx
6456 0000E2C3 0FB705[569D0100]    <1>     movzx   eax, word [v_width]
6457 0000E2CA 89C3                <1>     mov     ebx, eax
6458                              <1>     ; 12/02/2021
6459 0000E2CC F7E2                <1>     mul     edx ; rows * [v_width]
6460 0000E2CE 01F8                <1>     add     eax, edi
6461 0000E2D0 3B05[669D0100]      <1>     cmp     eax, [v_end]
6462 0000E2D6 58                  <1>     pop     eax ; color
6463 0000E2D7 7717                <1>     ja      short pix_op_lin_err1 ; out of display page
6464                              <1>     ; ecx = rows
6465 0000E2D9 89CA                <1>     mov     edx, ecx
6466                              <1>
6467 0000E2DB 803D[599D0100]08    <1>     cmp     byte [v_bpp], 8 ; 8bpp
6468 0000E2E2 770D                <1>     ja      short pix_op_lin_v_2
6469                              <1>     ; 256 colors (1 byte per pixel)
6470 0000E2E4 010D[348E0100]      <1>     add     [u.r0], ecx ; byte count
6471                              <1> pix_op_lin_v_1:
6472 0000E2EA 8807                <1>     mov     [edi], al
6473 0000E2EC 01DF                <1>     add     edi, ebx  ; next row
6474 0000E2EE E2FA                <1>     loop    pix_op_lin_v_1
6475                              <1> pix_op_lin_err1:
6476 0000E2F0 C3                  <1>     retn
6477                              <1>
6478                              <1> pix_op_lin_v_2:
6479 0000E2F1 803D[599D0100]18    <1>     cmp     byte [v_bpp], 24 ; 24bpp
6480 0000E2F8 773A                <1>     ja      short pix_op_lin_v_6 ; 32bpp
6481 0000E2FA 7226                <1>     jb      short pix_op_lin_v_4 ; 16bpp
6482                              <1>
6483                              <1>     ; 24 bit true colors
6484                              <1>     ; * 3
6485 0000E2FC 53                  <1>     push    ebx ; screen width in pixels
6486 0000E2FD D1E3                <1>     shl     ebx, 1
6487 0000E2FF 011C24              <1>     add     [esp], ebx
6488 0000E302 5B                  <1>     pop     ebx ; screen width in bytes
6489 0000E303 010D[348E0100]      <1>     add     [u.r0], ecx
6490 0000E309 D1E2                <1>     shl     edx, 1
6491 0000E30B 0115[348E0100]      <1>     add     [u.r0], edx ; byte count
6492                              <1> pix_op_lin_v_3:
6493 0000E311 668907              <1>     mov     [edi], ax
6494 0000E314 C1C810              <1>     ror     eax, 16
6495 0000E317 884702              <1>     mov     [edi+2], al
6496 0000E31A C1C010              <1>     rol     eax, 16
6497 0000E31D 01DF                <1>     add     edi, ebx  ; next row
6498 0000E31F E2F0                <1>     loop    pix_op_lin_v_3
6499 0000E321 C3                  <1>     retn
6500                              <1>
6501                              <1> pix_op_lin_v_4:
6502                              <1>     ; 16 bit (65536) colors
6503 0000E322 D1E3                <1>     shl     ebx, 1
6504 0000E324 D1E2                <1>     shl     edx, 1
6505 0000E326 0115[348E0100]      <1>     add     [u.r0], edx
6506                              <1> pix_op_lin_v_5:
6507 0000E32C 668907              <1>     mov     [edi], ax
6508 0000E32F 01DF                <1>     add     edi, ebx  ; next row
6509 0000E331 E2F9                <1>     loop    pix_op_lin_v_5
6510 0000E333 C3                  <1>     retn
6511                              <1>
6512                              <1> pix_op_lin_v_6:
6513                              <1>     ; 32 bit true colors
6514 0000E334 C1E302              <1>     shl     ebx, 2
6515 0000E337 C1E202              <1>     shl     edx, 2
6516 0000E33A 0115[348E0100]      <1>     add     [u.r0], edx ; byte count
6517                              <1> pix_op_lin_v_7:
6518 0000E340 8907                <1>     mov     [edi], eax
6519 0000E342 01DF                <1>     add     edi, ebx  ; next row
6520 0000E344 E2FA                <1>     loop    pix_op_lin_v_7
6521 0000E346 C3                  <1>     retn
6522                              <1>
6523                              <1> pix_op_lin_h:
6524                              <1>     ; Horizontal line
6525 0000E347 80E60F              <1>     and     dh, 0Fh ; low 12 bits
6526 0000E34A 89D1                <1>     mov     ecx, edx
6527 0000E34C 6601D6              <1>     add     si, dx ; start column + columns
6528 0000E34F 663B35[569D0100]    <1>     cmp     si, [v_width] ; screen width
6529 0000E356 7711                <1>     ja      short pix_op_lin_err2 ; out of columns limit
6530                              <1>
6531 0000E358 803D[599D0100]08    <1>     cmp     byte [v_bpp], 8 ; 8bpp
6532 0000E35F 7709                <1>     ja      short pix_op_lin_h_1
6533                              <1>     ; 256 colors (1 byte per pixel)
6534 0000E361 010D[348E0100]      <1>     add     [u.r0], ecx
```

```
6535 0000E367 F3AA                <1>     rep     stosb
6536                              <1> pix_op_lin_err2:
6537 0000E369 C3                  <1>     retn
6538                              <1>
6539                              <1> pix_op_lin_h_1:
6540 0000E36A 803D[599D0100]18    <1>     cmp     byte [v_bpp], 24 ; 24bpp
6541 0000E371 7728                <1>     ja      short pix_op_lin_h_4 ; 32bpp
6542 0000E373 721A                <1>     jb      short pix_op_lin_h_3 ; 16bpp
6543                              <1>
6544                              <1>     ; 24 bit true colors
6545                              <1>     ; * 3
6546 0000E375 0115[348E0100]      <1>     add     [u.r0], edx
6547 0000E37B D1E2                <1>     shl     edx, 1
6548 0000E37D 0115[348E0100]      <1>     add     [u.r0], edx
6549                              <1> pix_op_lin_h_2:
6550 0000E383 66AB                <1>     stosw
6551 0000E385 C1C810              <1>     ror     eax, 16
6552 0000E388 AA                  <1>     stosb
6553 0000E389 C1C010              <1>     rol     eax, 16
6554 0000E38C E2F5                <1>     loop    pix_op_lin_h_2
6555 0000E38E C3                  <1>     retn
6556                              <1>
6557                              <1> pix_op_lin_h_3:
6558                              <1>     ; 16 bit (65536) colors
6559 0000E38F D1E2                <1>     shl     edx, 1
6560 0000E391 0115[348E0100]      <1>     add     [u.r0], edx
6561 0000E397 F366AB              <1>     rep     stosw
6562 0000E39A C3                  <1>     retn
6563                              <1>
6564                              <1> pix_op_lin_h_4:
6565                              <1>     ; 32 bit true colors
6566 0000E39B C1E202              <1>     shl     edx, 2
6567 0000E39E 0115[348E0100]      <1>     add     [u.r0], edx
6568 0000E3A4 F3AB                <1>     rep     stosd
6569 0000E3A6 C3                  <1>     retn
6570                              <1>
6571                              <1> pix_op_new_8:
6572                              <1>     ; 8 bit colors (256 colors)
6573                              <1>     ; CHANGE PIXEL COLOR
6574                              <1>     ; ecx = pixel count per row
6575                              <1>     ;  al = color
6576                              <1>     ; edi = start pixel address
6577                              <1>
6578 0000E3A7 F3AA                <1>     rep     stosb
6579 0000E3A9 C3                  <1>     retn
6580                              <1>
6581                              <1> pix_op_new_16:
6582                              <1>     ; 16 bit colors (65536 colors)
6583                              <1>     ; CHANGE PIXEL COLOR
6584                              <1>     ; ecx = pixel count per row
6585                              <1>     ;  ax = color
6586                              <1>     ; edi = start pixel address
6587                              <1>
6588 0000E3AA F366AB              <1>     rep     stosw
6589 0000E3AD C3                  <1>     retn
6590                              <1>
6591                              <1> pix_op_new_24:
6592                              <1>     ; 24 bit true colors
6593                              <1>     ; CHANGE PIXEL COLOR
6594                              <1>     ; ecx = pixel count per row
6595                              <1>     ; eax = color
6596                              <1>     ; edi = start pixel address
6597                              <1>
6598 0000E3AE 66AB                <1>     stosw
6599 0000E3B0 C1C810              <1>     ror     eax, 16
6600 0000E3B3 AA                  <1>     stosb
6601 0000E3B4 C1C010              <1>     rol     eax, 16
6602 0000E3B7 E2F5                <1>     loop    pix_op_new_24
6603 0000E3B9 C3                  <1>     retn
6604                              <1>
6605                              <1> pix_op_new_32:
6606                              <1>     ; 32 bit true colors
6607                              <1>     ; CHANGE PIXEL COLOR
6608                              <1>     ; ecx = pixel count per row
6609                              <1>     ; eax = color
6610                              <1>     ; edi = start pixel address
6611                              <1>
6612 0000E3BA F3AB                <1>     rep     stosd
6613 0000E3BC C3                  <1>     retn
6614                              <1>
6615                              <1> pix_op_add_8:
6616                              <1>     ; 8 bit colors (256 colors)
6617                              <1>     ; ADD PIXEL COLOR
6618                              <1>     ; ecx = pixel count per row
6619                              <1>     ;  al = color
6620                              <1>     ; edi = start pixel address
6621                              <1>
6622 0000E3BD 88C4                <1>     mov     ah, al
6623                              <1> pix_op_add_8_0:
6624 0000E3BF 0207                <1>     add     al, [edi]
6625 0000E3C1 7302                <1>     jnc     short pix_op_add_8_1
6626 0000E3C3 B0FF                <1>     mov     al, 0FFh ; Max. value
6627                              <1> pix_op_add_8_1:
6628 0000E3C5 AA                  <1>     stosb
6629 0000E3C6 88E0                <1>     mov     al, ah
6630 0000E3C8 E2F5                <1>     loop    pix_op_add_8_0
6631 0000E3CA C3                  <1>     retn
6632                              <1>
6633                              <1> pix_op_add_16:
6634                              <1>     ; 16 bit colors (65536 colors)
6635                              <1>     ; ADD PIXEL COLOR
6636                              <1>     ; ecx = pixel count per row
6637                              <1>     ;  ax = color
6638                              <1>     ; edi = start pixel address
6639                              <1>
6640 0000E3CB 89C2                <1>     mov     edx, eax
6641                              <1> pix_op_add_16_0:
6642 0000E3CD 660307              <1>     add     ax, [edi]
6643 0000E3D0 7304                <1>     jnc     short pix_op_add_16_1
6644 0000E3D2 66B8FFFF            <1>     mov     ax, 0FFFFh ; Max. value
6645                              <1> pix_op_add_16_1:
6646 0000E3D6 66AB                <1>     stosw
6647 0000E3D8 89D0                <1>     mov     eax, edx
6648 0000E3DA E2F1                <1>     loop    pix_op_add_16_0
6649 0000E3DC C3                  <1>     retn
6650                              <1>
6651                              <1> pix_op_add_24:
6652                              <1>     ; 24 bit true colors
6653                              <1>     ; ADD PIXEL COLOR
6654                              <1>     ; ecx = pixel count per row
6655                              <1>     ; eax = color
6656                              <1>     ; edi = start pixel address
6657                              <1>
6658 0000E3DD 53                  <1>     push    ebx
```

```
6659 0000E3DE BBFFFFFF00          <1>     mov     ebx, 0FFFFFFh
6660                              <1>     ;and    eax, ebx ; 0FFFFFFh
6661 0000E3E3 89C2                <1>     mov     edx, eax
6662                              <1> pix_op_add_24_0:
6663 0000E3E5 8B07                <1>     mov     eax, [edi]
6664 0000E3E7 21D8                <1>     and     eax, ebx ; 0FFFFFFh
6665 0000E3E9 01D0                <1>     add     eax, edx
6666 0000E3EB 39D8                <1>     cmp     eax, ebx
6667 0000E3ED 7602                <1>     jna     short pix_op_add_24_1
6668 0000E3EF 89D8                <1>     mov     eax, ebx ; 0FFFFFFh  ; Max. value
6669                              <1> pix_op_add_24_1:
6670 0000E3F1 66AB                <1>     stosw
6671 0000E3F3 C1E810              <1>     shr     eax, 16
6672 0000E3F6 AA                  <1>     stosb
6673 0000E3F7 E2EC                <1>     loop    pix_op_add_24_0
6674 0000E3F9 89D0                <1>     mov     eax, edx
6675 0000E3FB 5B                  <1>     pop     ebx
6676 0000E3FC C3                  <1>     retn
6677                              <1>
6678                              <1> pix_op_add_32:
6679                              <1>     ; 32 bit true colors
6680                              <1>     ; ADD PIXEL COLOR
6681                              <1>     ; ecx = pixel count per row
6682                              <1>     ; eax = color
6683                              <1>     ; edi = start pixel address
6684                              <1>
6685 0000E3FD 89C2                <1>     mov     edx, eax
6686                              <1> pix_op_add_32_0:
6687 0000E3FF 0307                <1>     add     eax, [edi]
6688 0000E401 7303                <1>     jnc     short pix_op_add_32_1
6689                              <1>     ;mov    eax, 0FFFFFFFFh ; Max. value
6690 0000E403 29C0                <1>     sub     eax, eax
6691 0000E405 48                  <1>     dec     eax
6692                              <1> pix_op_add_32_1:
6693 0000E406 AB                  <1>     stosd
6694 0000E407 89D0                <1>     mov     eax, edx
6695 0000E409 E2F4                <1>     loop    pix_op_add_32_0
6696 0000E40B C3                  <1>     retn
6697                              <1>
6698                              <1> pix_op_sub_8:
6699                              <1>     ; 8 bit colors (256 colors)
6700                              <1>     ; SUBTRACT PIXEL COLOR
6701                              <1>     ; ecx = pixel count per row
6702                              <1>     ;  al = color
6703                              <1>     ; edi = start pixel address
6704                              <1>
6705 0000E40C 88C4                <1>     mov     ah, al
6706                              <1> pix_op_sub_8_0:
6707 0000E40E 8A07                <1>     mov     al, [edi]
6708 0000E410 28E0                <1>     sub     al, ah
6709 0000E412 7302                <1>     jnb     short pix_op_sub_8_1
6710 0000E414 30C0                <1>     xor     al, al ; 0 ; Min. value
6711                              <1> pix_op_sub_8_1:
6712 0000E416 AA                  <1>     stosb
6713 0000E417 E2F5                <1>     loop    pix_op_sub_8_0
6714 0000E419 88E0                <1>     mov     al, ah
6715 0000E41B C3                  <1>     retn
6716                              <1>
6717                              <1> pix_op_sub_16:
6718                              <1>     ; 16 bit colors (65536 colors)
6719                              <1>     ; SUBTRACT PIXEL COLOR
6720                              <1>     ; ecx = pixel count per row
6721                              <1>     ;  ax = color
6722                              <1>     ; edi = start pixel address
6723                              <1>
6724 0000E41C 89C2                <1>     mov     edx, eax
6725                              <1> pix_op_sub_16_0:
6726 0000E41E 668B07              <1>     mov     ax, [edi]
6727 0000E421 6629D0              <1>     sub     ax, dx
6728 0000E424 7302                <1>     jnb     short pix_op_sub_16_1
6729 0000E426 31C0                <1>     xor     eax, eax ; 0 ; Min. value
6730                              <1> pix_op_sub_16_1:
6731 0000E428 66AB                <1>     stosw
6732 0000E42A E2F2                <1>     loop    pix_op_sub_16_0
6733 0000E42C 89D0                <1>     mov     eax, edx
6734 0000E42E C3                  <1>     retn
6735                              <1>
6736                              <1> pix_op_sub_24:
6737                              <1>     ; 24 bit true colors
6738                              <1>     ; SUBTRACT PIXEL COLOR
6739                              <1>     ; ecx = pixel count per row
6740                              <1>     ; eax = color
6741                              <1>     ; edi = start pixel address
6742                              <1>
6743                              <1>     ;and    eax, 0FFFFFFh
6744 0000E42F 89C2                <1>     mov     edx, eax
6745                              <1> pix_op_sub_24_0:
6746 0000E431 8B07                <1>     mov     eax, [edi]
6747                              <1>     ; 27/02/2021
6748 0000E433 25FFFFFF00          <1>     and     eax, 0FFFFFFh
6749 0000E438 29D0                <1>     sub     eax, edx
6750 0000E43A 7302                <1>     jnb     short pix_op_sub_24_1
6751 0000E43C 31C0                <1>     xor     eax, eax ; 0 ; Min. value
6752                              <1> pix_op_sub_24_1:
6753 0000E43E 66AB                <1>     stosw
6754 0000E440 C1E810              <1>     shr     eax, 16
6755 0000E443 AA                  <1>     stosb
6756 0000E444 E2EB                <1>     loop    pix_op_sub_24_0
6757 0000E446 89D0                <1>     mov     eax, edx
6758 0000E448 C3                  <1>     retn
6759                              <1>
6760                              <1> pix_op_sub_32:
6761                              <1>     ; 32 bit true colors
6762                              <1>     ; SUBTRACT PIXEL COLOR
6763                              <1>     ; ecx = pixel count per row
6764                              <1>     ; eax = color
6765                              <1>     ; edi = start pixel address
6766                              <1>
6767 0000E449 89C2                <1>     mov     edx, eax
6768                              <1> pix_op_sub_32_0:
6769 0000E44B 8B07                <1>     mov     eax, [edi]
6770 0000E44D 29D0                <1>     sub     eax, edx
6771 0000E44F 7302                <1>     jnb     short pix_op_sub_32_1
6772 0000E451 31C0                <1>     xor     eax, eax ; 0 ; Min. value
6773                              <1> pix_op_sub_32_1:
6774 0000E453 AB                  <1>     stosd
6775 0000E454 E2F5                <1>     loop    pix_op_sub_32_0
6776 0000E456 89D0                <1>     mov     eax, edx
6777 0000E458 C3                  <1>     retn
6778                              <1>
6779                              <1> pix_op_or_8:
6780                              <1>     ; 8 bit colors (256 colors)
6781                              <1>     ; OR PIXEL COLOR
6782                              <1>     ; ecx = pixel count per row
```

```
                            <1>     ;  al = color
                            <1>     ; edi = start pixel address
                            <1>
                            <1> pix_op_or_8_0:
0000E459 0807               <1>     or      [edi], al
0000E45B 47                 <1>     inc     edi
0000E45C E2FB               <1>     loop    pix_op_or_8_0
0000E45E C3                 <1>     retn
                            <1>
                            <1> pix_op_or_16:
                            <1>     ; 16 bit colors (65536 colors)
                            <1>     ; OR PIXEL COLOR
                            <1>     ; ecx = pixel count per row
                            <1>     ;  ax = color
                            <1>     ; edi = start pixel address
                            <1>
                            <1> pix_op_or_16_0:
0000E45F 660907             <1>     or      [edi], ax
0000E462 47                 <1>     inc     edi
0000E463 47                 <1>     inc     edi
0000E464 E2F9               <1>     loop    pix_op_or_16_0
0000E466 C3                 <1>     retn
                            <1>
                            <1> pix_op_or_24:
                            <1>     ; 24 bit true colors
                            <1>     ; OR PIXEL COLOR
                            <1>     ; ecx = pixel count per row
                            <1>     ; eax = color
                            <1>     ; edi = start pixel address
                            <1>
0000E467 89C2               <1>     mov     edx, eax
                            <1> pix_op_or_24_0:
0000E469 0B07               <1>     or      eax, [edi]
0000E46B 66AB               <1>     stosw
0000E46D C1E810             <1>     shr     eax, 16
0000E470 AA                 <1>     stosb
0000E471 89D0               <1>     mov     eax, edx
0000E473 E2F4               <1>     loop    pix_op_or_24_0
0000E475 C3                 <1>     retn
                            <1>
                            <1> pix_op_or_32:
                            <1>     ; 32 bit true colors
                            <1>     ; OR PIXEL COLOR
                            <1>     ; ecx = pixel count per row
                            <1>     ; eax = color
                            <1>     ; edi = start pixel address
                            <1>
                            <1>     ;mov    edx, eax
                            <1> pix_op_or_32_0:
                            <1>     ;or     eax, [edi]
                            <1>     ;stosd
                            <1>     ;mov    eax, edx
0000E476 0907               <1>     or      [edi], eax
0000E478 83C704             <1>     add     edi, 4
0000E47B E2F9               <1>     loop    pix_op_or_32_0
0000E47D C3                 <1>     retn
                            <1>
                            <1> pix_op_and_8:
                            <1>     ; 8 bit colors (256 colors)
                            <1>     ; AND PIXEL COLOR
                            <1>     ; ecx = pixel count per row
                            <1>     ;  al = color
                            <1>     ; edi = start pixel address
                            <1>
                            <1> pix_op_and_8_0:
0000E47E 2007               <1>     and     [edi], al
0000E480 47                 <1>     inc     edi
0000E481 E2FB               <1>     loop    pix_op_and_8_0
0000E483 C3                 <1>     retn
                            <1>
                            <1> pix_op_and_16:
                            <1>     ; 16 bit colors (65536 colors)
                            <1>     ; AND PIXEL COLOR
                            <1>     ; ecx = pixel count per row
                            <1>     ;  ax = color
                            <1>     ; edi = start pixel address
                            <1>
                            <1> pix_op_and_16_0:
0000E484 662107             <1>     and     [edi], ax
0000E487 47                 <1>     inc     edi
0000E488 47                 <1>     inc     edi
0000E489 E2F9               <1>     loop    pix_op_and_16_0
0000E48B C3                 <1>     retn
                            <1>
                            <1> pix_op_and_24:
                            <1>     ; 24 bit true colors
                            <1>     ; AND PIXEL COLOR
                            <1>     ; ecx = pixel count per row
                            <1>     ; eax = color
                            <1>     ; edi = start pixel address
                            <1>
0000E48C 89C2               <1>     mov     edx, eax
                            <1> pix_op_and_24_0:
0000E48E 2307               <1>     and     eax, [edi]
0000E490 66AB               <1>     stosw
0000E492 C1E810             <1>     shr     eax, 16
0000E495 AA                 <1>     stosb
0000E496 89D0               <1>     mov     eax, edx
0000E498 E2F4               <1>     loop    pix_op_and_24_0
0000E49A C3                 <1>     retn
                            <1>
                            <1> pix_op_and_32:
                            <1>     ; 32 bit true colors
                            <1>     ; AND PIXEL COLOR
                            <1>     ; ecx = pixel count per row
                            <1>     ; eax = color
                            <1>     ; edi = start pixel address
                            <1>
                            <1>     ;mov    edx, eax
                            <1> pix_op_and_32_0:
                            <1>     ;and    eax, [edi]
                            <1>     ;stosd
                            <1>     ;mov    eax, edx
0000E49B 2107               <1>     and     [edi], eax
0000E49D 83C704             <1>     add     edi, 4
0000E4A0 E2F9               <1>     loop    pix_op_and_32_0
0000E4A2 C3                 <1>     retn
                            <1>
                            <1> pix_op_xor_8:
                            <1>     ; 8 bit colors (256 colors)
                            <1>     ; XOR PIXEL COLOR
                            <1>     ; ecx = pixel count per row
                            <1>     ;  al = color
                            <1>     ; edi = start pixel address
```

```
                                <1>
                                <1> pix_op_xor_8_0:
0000E4A3 3007                   <1>     xor     [edi], al
0000E4A5 47                     <1>     inc     edi
0000E4A6 E2FB                   <1>     loop    pix_op_xor_8_0
0000E4A8 C3                     <1>     retn
                                <1>
                                <1> pix_op_xor_16:
                                <1>     ; 16 bit colors (65536 colors)
                                <1>     ; XOR PIXEL COLOR
                                <1>     ; ecx = pixel count per row
                                <1>     ;  ax = color
                                <1>     ; edi = start pixel address
                                <1>
                                <1> pix_op_xor_16_0:
0000E4A9 663107                 <1>     xor     [edi], ax
0000E4AC 47                     <1>     inc     edi
0000E4AD 47                     <1>     inc     edi
0000E4AE E2F9                   <1>     loop    pix_op_xor_16_0
0000E4B0 C3                     <1>     retn
                                <1>
                                <1> pix_op_xor_24:
                                <1>     ; 24 bit true colors
                                <1>     ; XOR PIXEL COLOR
                                <1>     ; ecx = pixel count per row
                                <1>     ; eax = color
                                <1>     ; edi = start pixel address
                                <1>
0000E4B1 89C2                   <1>     mov     edx, eax
                                <1> pix_op_xor_24_0:
0000E4B3 3307                   <1>     xor     eax, [edi]
0000E4B5 66AB                   <1>     stosw
0000E4B7 C1E810                 <1>     shr     eax, 16
0000E4BA AA                     <1>     stosb
0000E4BB 89D0                   <1>     mov     eax, edx
0000E4BD E2F4                   <1>     loop    pix_op_xor_24_0
0000E4BF C3                     <1>     retn
                                <1>
                                <1> pix_op_xor_32:
                                <1>     ; 32 bit true colors
                                <1>     ; XOR PIXEL COLOR
                                <1>     ; ecx = pixel count per row
                                <1>     ; eax = color
                                <1>     ; edi = start pixel address
                                <1>
                                <1>     ;mov    edx, eax
                                <1> pix_op_xor_32_0:
                                <1>     ;xor    eax, [edi]
                                <1>     ;stosd
                                <1>     ;mov    eax, edx
0000E4C0 3107                   <1>     xor     [edi], eax
0000E4C2 83C704                 <1>     add     edi, 4
0000E4C5 E2F9                   <1>     loop    pix_op_xor_32_0
0000E4C7 C3                     <1>     retn
                                <1>
                                <1> pix_op_mix_8:
                                <1>     ; 8 bit colors (256 colors)
                                <1>     ; MIX (AVERAGE) PIXEL COLORS
                                <1>     ; ecx = pixel count per row
                                <1>     ;  al = color
                                <1>     ; edi = start pixel address
                                <1>
0000E4C8 88C4                   <1>     mov     ah, al
                                <1> pix_op_mix_8_0:
0000E4CA 0207                   <1>     add     al, [edi]
0000E4CC D0D8                   <1>     rcr     al, 1
0000E4CE AA                     <1>     stosb
0000E4CF 88E0                   <1>     mov     al, ah
0000E4D1 E2F7                   <1>     loop    pix_op_mix_8_0
0000E4D3 C3                     <1>     retn
                                <1>
                                <1> pix_op_mix_16:
                                <1>     ; 16 bit colors (65536 colors)
                                <1>     ; MIX (AVERAGE) PIXEL COLORS
                                <1>     ; ecx = pixel count per row
                                <1>     ;  ax = color
                                <1>     ; edi = start pixel address
                                <1>
0000E4D4 89C2                   <1>     mov     edx, eax
                                <1> pix_op_mix_16_0:
0000E4D6 660307                 <1>     add     ax, [edi]
0000E4D9 66D1D8                 <1>     rcr     ax, 1
0000E4DC 66AB                   <1>     stosw
0000E4DE 89D0                   <1>     mov     eax, edx
0000E4E0 E2F4                   <1>     loop    pix_op_mix_16_0
0000E4E2 C3                     <1>     retn
                                <1>
                                <1> pix_op_mix_24:
                                <1>     ; 24 bit true colors
                                <1>     ; MIX (AVERAGE) PIXEL COLORS
                                <1>     ; ecx = pixel count per row
                                <1>     ; eax = color
                                <1>     ; edi = start pixel address
                                <1>
0000E4E3 53                     <1>     push    ebx
0000E4E4 BBFFFFFF00             <1>     mov     ebx, 0FFFFFFh
                                <1>     ;and    eax, ebx ; 0FFFFFFh
0000E4E9 89C2                   <1>     mov     edx, eax
                                <1> pix_op_mix_24_0:
0000E4EB 8B07                   <1>     mov     eax, [edi]
0000E4ED 21D8                   <1>     and     eax, ebx ; 0FFFFFFh
0000E4EF 01D0                   <1>     add     eax, edx
0000E4F1 D1E8                   <1>     shr     eax, 1
                                <1>     ;rcr    eax, 1
0000E4F3 66AB                   <1>     stosw
0000E4F5 C1E810                 <1>     shr     eax, 16
0000E4F8 AA                     <1>     stosb
0000E4F9 E2F0                   <1>     loop    pix_op_mix_24_0
0000E4FB 89D0                   <1>     mov     eax, edx
0000E4FD 5B                     <1>     pop     ebx
0000E4FE C3                     <1>     retn
                                <1>
                                <1> pix_op_mix_32:
                                <1>     ; 32 bit true colors
                                <1>     ; MIX (AVERAGE) PIXEL COLORS
                                <1>     ; ecx = pixel count per row
                                <1>     ; eax = color
                                <1>     ; edi = start pixel address
                                <1>
0000E4FF 89C2                   <1>     mov     edx, eax
                                <1> pix_op_mix_32_0:
0000E501 0307                   <1>     add     eax, [edi]
0000E503 D1D8                   <1>     rcr     eax, 1
0000E505 AB                     <1>     stosd
```

```
7031 0000E506 89D0              <1>     mov     eax, edx
7032 0000E508 E2F7              <1>     loop    pix_op_mix_32_0
7033 0000E50A C3                <1>     retn
7034                            <1>
7035                            <1> pix_op_not_8:
7036                            <1>     ; 8 bit colors (256 colors)
7037                            <1>     ; NOT PIXEL COLOR
7038                            <1>     ; ecx = pixel count per row
7039                            <1>     ; edi = start pixel address
7040                            <1>
7041                            <1> pix_op_not_8_0:
7042 0000E50B F617              <1>     not     byte [edi]
7043 0000E50D 47                <1>     inc     edi
7044 0000E50E E2FB              <1>     loop    pix_op_not_8_0
7045 0000E510 C3                <1>     retn
7046                            <1>
7047                            <1> pix_op_not_16:
7048                            <1>     ; 16 bit colors (65536 colors)
7049                            <1>     ; NOT PIXEL COLOR
7050                            <1>     ; ecx = pixel count per row
7051                            <1>     ; edi = start pixel address
7052                            <1>
7053                            <1> pix_op_not_16_0:
7054 0000E511 66F717            <1>     not     word [edi]
7055 0000E514 47                <1>     inc     edi
7056 0000E515 47                <1>     inc     edi
7057 0000E516 E2F9              <1>     loop    pix_op_not_16_0
7058 0000E518 C3                <1>     retn
7059                            <1>
7060                            <1> pix_op_not_24:
7061                            <1>     ; 24 bit true colors
7062                            <1>     ; NOT PIXEL COLOR
7063                            <1>     ; ecx = pixel count per row
7064                            <1>     ; edi = start pixel address
7065                            <1>
7066                            <1> pix_op_not_24_0:
7067 0000E519 66F717            <1>     not     word [edi]
7068 0000E51C 47                <1>     inc     edi
7069 0000E51D 47                <1>     inc     edi
7070 0000E51E F617              <1>     not     byte [edi]
7071 0000E520 47                <1>     inc     edi
7072 0000E521 E2F6              <1>     loop    pix_op_not_24_0
7073 0000E523 C3                <1>     retn
7074                            <1>
7075                            <1> pix_op_not_32:
7076                            <1>     ; 32 bit true colors
7077                            <1>     ; NOT PIXEL COLOR
7078                            <1>     ; ecx = pixel count per row
7079                            <1>     ; eax = color
7080                            <1>     ; edi = start pixel address
7081                            <1> pix_op_not_32_0:
7082 0000E524 F717              <1>     not     dword [edi]
7083 0000E526 83C704            <1>     add     edi, 4
7084 0000E529 E2F9              <1>     loop    pix_op_not_32_0
7085 0000E52B C3                <1>     retn
7086                            <1>
7087                            <1> pix_op_neg_8:
7088                            <1>     ; 8 bit colors (256 colors)
7089                            <1>     ; NEG PIXEL COLOR
7090                            <1>     ; ecx = pixel count per row
7091                            <1>     ; edi = start pixel address
7092                            <1>
7093                            <1> pix_op_neg_8_0:
7094 0000E52C F61F              <1>     neg     byte [edi]
7095 0000E52E 47                <1>     inc     edi
7096 0000E52F E2FB              <1>     loop    pix_op_neg_8_0
7097 0000E531 C3                <1>     retn
7098                            <1>
7099                            <1> pix_op_neg_16:
7100                            <1>     ; 16 bit colors (65536 colors)
7101                            <1>     ; NEG PIXEL COLOR
7102                            <1>     ; ecx = pixel count per row
7103                            <1>     ; edi = start pixel address
7104                            <1>
7105                            <1> pix_op_neg_16_0:
7106 0000E532 66F71F            <1>     neg     word [edi]
7107 0000E535 47                <1>     inc     edi
7108 0000E536 47                <1>     inc     edi
7109 0000E537 E2F9              <1>     loop    pix_op_neg_16_0
7110 0000E539 C3                <1>     retn
7111                            <1>
7112                            <1> pix_op_neg_24:
7113                            <1>     ; 24 bit true colors
7114                            <1>     ; NEG PIXEL COLOR
7115                            <1>     ; ecx = pixel count per row
7116                            <1>     ; edi = start pixel address
7117                            <1>
7118                            <1> pix_op_neg_24_0:
7119 0000E53A 8B07              <1>     mov     eax, [edi]
7120 0000E53C 25FFFFFF00        <1>     and     eax, 0FFFFFFh
7121 0000E541 F7D8              <1>     neg     eax
7122 0000E543 66AB              <1>     stosw
7123 0000E545 C1E810            <1>     shr     eax, 16
7124 0000E548 AA                <1>     stosb
7125 0000E549 E2EF              <1>     loop    pix_op_neg_24_0
7126 0000E54B C3                <1>     retn
7127                            <1>
7128                            <1> pix_op_neg_32:
7129                            <1>     ; 32 bit true colors
7130                            <1>     ; NEG PIXEL COLOR
7131                            <1>     ; ecx = pixel count per row
7132                            <1>     ; eax = color
7133                            <1>     ; edi = start pixel address
7134                            <1> pix_op_neg_32_0:
7135 0000E54C F71F              <1>     neg     dword [edi]
7136 0000E54E 83C704            <1>     add     edi, 4
7137 0000E551 E2F9              <1>     loop    pix_op_neg_32_0
7138 0000E553 C3                <1>     retn
7139                            <1>
7140                            <1> pix_op_inc_8:
7141                            <1>     ; 8 bit colors (256 colors)
7142                            <1>     ; INCREASE PIXEL COLOR
7143                            <1>     ; ecx = pixel count per row
7144                            <1>     ; edi = start pixel address
7145                            <1>
7146                            <1> pix_op_inc_8_0:
7147 0000E554 FE07              <1>     inc     byte [edi]
7148 0000E556 7502              <1>     jnz     short pix_op_inc_8_1
7149                            <1>     ;mov    [edi], 0FFh ; Max. value
7150 0000E558 FE0F              <1>     dec     byte [edi]
7151                            <1> pix_op_inc_8_1:
7152 0000E55A 47                <1>     inc     edi
7153 0000E55B E2F7              <1>     loop    pix_op_inc_8_0
7154 0000E55D C3                <1>     retn
```

```
                                <1>
                                <1> pix_op_inc_16:
                                <1>     ; 16 bit colors (65536 colors)
                                <1>     ; INCREASE PIXEL COLOR
                                <1>     ; ecx = pixel count per row
                                <1>     ; edi = start pixel address
                                <1>
                                <1> pix_op_inc_16_0:
0000E55E 66FF07                 <1>     inc     word [edi]
0000E561 7503                   <1>     jnz     short pix_op_inc_16_1
                                <1>     ;mov    word [edi], 0FFFFh ; Max. value
0000E563 66FF0F                 <1>     dec     word [edi]
                                <1> pix_op_inc_16_1:
0000E566 47                     <1>     inc     edi
0000E567 47                     <1>     inc     edi
0000E568 E2F4                   <1>     loop    pix_op_inc_16_0
0000E56A C3                     <1>     retn
                                <1>
                                <1> pix_op_inc_24:
                                <1>     ; 24 bit true colors
                                <1>     ; INCREASE PIXEL COLOR
                                <1>     ; ecx = pixel count per row
                                <1>     ; edi = start pixel address
                                <1>
                                <1> pix_op_inc_24_0:
0000E56B 8B07                   <1>     mov     eax, [edi]
0000E56D 40                     <1>     inc     eax
0000E56E 25FFFFFF00             <1>     and     eax, 0FFFFFFh
0000E573 7501                   <1>     jnz     short pix_op_inc_24_1
                                <1>     ;mov    eax, 0FFFFFFh  ; Max. value
0000E575 48                     <1>     dec     eax ; 0FFFFFFFFh
                                <1> pix_op_inc_24_1:
0000E576 66AB                   <1>     stosw
0000E578 C1E810                 <1>     shr     eax, 16
0000E57B AA                     <1>     stosb
0000E57C E2ED                   <1>     loop    pix_op_inc_24_0
0000E57E C3                     <1>     retn
                                <1>
                                <1> pix_op_inc_32:
                                <1>     ; 32 bit true colors
                                <1>     ; INCREASE PIXEL COLOR
                                <1>     ; ecx = pixel count per row
                                <1>     ; edi = start pixel address
                                <1>
                                <1> pix_op_inc_32_0:
0000E57F FF07                   <1>     inc     dword [edi]
0000E581 7502                   <1>     jnz     short pix_op_inc_32_1
                                <1>     ;mov    dword [edi], 0FFFFFFFFh ; Max. value
0000E583 FF0F                   <1>     dec     dword [edi]
                                <1> pix_op_inc_32_1:
0000E585 83C704                 <1>     add     edi, 4
0000E588 E2F5                   <1>     loop    pix_op_inc_32_0
0000E58A C3                     <1>     retn
                                <1>
                                <1> pix_op_dec_8:
                                <1>     ; 8 bit colors (256 colors)
                                <1>     ; DECREASE PIXEL COLOR
                                <1>     ; ecx = pixel count per row
                                <1>     ; edi = start pixel address
                                <1>
                                <1> pix_op_dec_8_0:
0000E58B FE0F                   <1>     dec     byte [edi]
0000E58D 7902                   <1>     jns     short pix_op_dec_8_1
0000E58F FE07                   <1>     inc     byte [edi] ; 0 ; Min. value
                                <1> pix_op_dec_8_1:
0000E591 47                     <1>     inc     edi
0000E592 E2F7                   <1>     loop    pix_op_dec_8_0
0000E594 C3                     <1>     retn
                                <1>
                                <1> pix_op_dec_16:
                                <1>     ; 16 bit colors (65536 colors)
                                <1>     ; DECREASE PIXEL COLOR
                                <1>     ; ecx = pixel count per row
                                <1>     ; edi = start pixel address
                                <1>
                                <1> pix_op_dec_16_0:
0000E595 66FF0F                 <1>     dec     word [edi]
0000E598 7903                   <1>     jns     short pix_op_dec_16_1
0000E59A 66FF07                 <1>     inc     word [edi] ; 0 ; Min. value
                                <1> pix_op_dec_16_1:
0000E59D 47                     <1>     inc edi
0000E59E 47                     <1>     inc edi
0000E59F E2F4                   <1>     loop    pix_op_dec_16_0
0000E5A1 C3                     <1>     retn
                                <1>
                                <1> pix_op_dec_24:
                                <1>     ; 24 bit true colors
                                <1>     ; DECREASE PIXEL COLOR
                                <1>     ; ecx = pixel count per row
                                <1>     ; edi = start pixel address
                                <1>
                                <1> pix_op_dec_24_0:
0000E5A2 8B07                   <1>     mov     eax, [edi]
0000E5A4 25FFFFFF00             <1>     and     eax, 0FFFFFFh
0000E5A9 7401                   <1>     jz      short pix_op_dec_24_1
                                <1>                     ; 0 ; Min. value
0000E5AB 48                     <1>     dec     eax
                                <1> pix_op_dec_24_1:
0000E5AC 66AB                   <1>     stosw
0000E5AE C1E810                 <1>     shr     eax, 16
0000E5B1 AA                     <1>     stosb
0000E5B2 E2B7                   <1>     loop    pix_op_inc_24_0
0000E5B4 C3                     <1>     retn
                                <1>
                                <1> pix_op_dec_32:
                                <1>     ; 32 bit true colors
                                <1>     ; DECREASE PIXEL COLOR
                                <1>     ; ecx = pixel count per row
                                <1>     ; edi = start pixel address
                                <1>
                                <1> pix_op_dec_32_0:
0000E5B5 FF0F                   <1>     dec     dword [edi]
0000E5B7 7902                   <1>     jns     short pix_op_dec_32_1
0000E5B9 FF07                   <1>     inc     dword [edi] ; 0 ; Min. value
                                <1> pix_op_dec_32_1:
0000E5BB 83C704                 <1>     add     edi, 4
0000E5BE E2F5                   <1>     loop    pix_op_dec_32_0
0000E5C0 89D0                   <1>     mov     eax, edx
0000E5C2 C3                     <1>     retn
                                <1>
                                <1> pix_op_rpl_8:
                                <1>     ; 8 bit colors (256 colors)
                                <1>     ; REPLACE PIXEL COLORS
                                <1>     ; ecx = pixel count per row
```

```
                                <1>     ;  al = new color
                                <1>     ; byte [maskcolor] = old color
                                <1>     ; edi = start pixel address
                                <1>
0000E5C3 8A25[6A9D0100]         <1>     mov     ah, [maskcolor]
                                <1> pix_op_rpl_8_0:
0000E5C9 3A27                   <1>     cmp     ah, [edi]
0000E5CB 7502                   <1>     jne     short pix_op_rpl_8_1
0000E5CD 8807                   <1>     mov     [edi], al
                                <1> pix_op_rpl_8_1:
0000E5CF 47                     <1>     inc     edi
0000E5D0 E2F7                   <1>     loop    pix_op_rpl_8_0
0000E5D2 C3                     <1>     retn
                                <1>
                                <1> pix_op_rpl_16:
                                <1>     ; 16 bit colors (65536 colors)
                                <1>     ; REPLACE PIXEL COLORS
                                <1>     ; ecx = pixel count per row
                                <1>     ;  ax = new color
                                <1>     ; word [maskcolor] = old color
                                <1>     ; edi = start pixel address
                                <1>
0000E5D3 8B15[6A9D0100]         <1>     mov     edx, [maskcolor]
                                <1> pix_op_rpl_16_0:
0000E5D9 663B17                 <1>     cmp     dx, [edi]
0000E5DC 7503                   <1>     jne     short pix_op_rpl_16_1
0000E5DE 668907                 <1>     mov     [edi], ax
                                <1> pix_op_rpl_16_1:
0000E5E1 47                     <1>     inc     edi
0000E5E2 47                     <1>     inc     edi
0000E5E3 E2F4                   <1>     loop    pix_op_rpl_16_0
0000E5E5 C3                     <1>     retn
                                <1>
                                <1> pix_op_rpl_24:
                                <1>     ; 24 bit true colors
                                <1>     ; REPLACE PIXEL COLORS
                                <1>     ; ecx = pixel count per row
                                <1>     ; eax = new color
                                <1>     ; [maskcolor] = old color
                                <1>     ; edi = start pixel address
                                <1>
                                <1> pix_op_rpl_24_0:
0000E5E6 8B17                   <1>     mov     edx, [edi]
0000E5E8 81E2FFFFFF00           <1>     and     edx, 0FFFFFFh
0000E5EE 3B15[6A9D0100]         <1>     cmp     edx, [maskcolor]
0000E5F4 7406                   <1>     je      short pix_op_rpl_24_1
0000E5F6 83C703                 <1>     add     edi, 3
0000E5F9 E2EB                   <1>     loop    pix_op_rpl_24_0
0000E5FB C3                     <1>     retn
                                <1> pix_op_rpl_24_1:
0000E5FC AA                     <1>     stosb
0000E5FD C1C808                 <1>     ror     eax, 8
0000E600 66AB                   <1>     stosw
0000E602 C1C008                 <1>     rol     eax, 8
0000E605 E2DF                   <1>     loop    pix_op_rpl_24_0
0000E607 C3                     <1>     retn
                                <1>
                                <1> pix_op_rpl_32:
                                <1>     ; 32 bit true colors
                                <1>     ; REPLACE PIXEL COLORS
                                <1>     ; ecx = pixel count per row
                                <1>     ; eax = new color
                                <1>     ; [maskcolor] = old color
                                <1>     ; edi = start pixel address
                                <1>
0000E608 8B15[6A9D0100]         <1>     mov     edx, [maskcolor]
                                <1> pix_op_rpl_32_0:
0000E60E 3B17                   <1>     cmp     edx, [edi]
0000E610 7504                   <1>     jne     short pix_op_rpl_32_2
0000E612 AB                     <1>     stosd
0000E613 E2F9                   <1>     loop    pix_op_rpl_32_0
0000E615 C3                     <1>     retn
                                <1> pix_op_rpl_32_2:
0000E616 83C704                 <1>     add     edi, 4
0000E619 E2F3                   <1>     loop    pix_op_rpl_32_0
0000E61B C3                     <1>     retn
                                <1>
                                <1> pix_op_chr:
                                <1>     ; 23/07/2022 - TRDOS 386 v2.0.5
                                <1>     ; 15/02/2021
                                <1>     ; 05/02/2021
                                <1>     ; WRITE CHARACTER (FONT)
                                <1>     ; 05/01/2021 ([ufont])
                                <1>     ; 01/01/2021
                                <1>     ;     CL = char's color (8 bit, 256 colors)
                                <1>     ;    ECX = char's color (16 bit and true colors)
                                <1>     ;     DL = Character's ASCII code
                                <1>     ;     DH bit 0 -> font height
                                <1>     ;        0 -> 8x16 character font
                                <1>     ;        1 -> 8x8 character font
                                <1>     ;     DH bit 1 & 2 -> scale
                                <1>     ;        0 = 1/1 (8 pixels per char row)
                                <1>     ;        1 = 2/1 (16 pixels per char row)
                                <1>     ;        2 = 3/1 (24 pixels per char row)
                                <1>     ;        3 = 4/1 (32 pixels per char row)
                                <1>     ;     DH bit 6 -> [ufont] option (1 = use [ufont])
                                <1>     ;     If DH bit 7 = 1
                                <1>     ;       USER FONT (from user buffer)
                                <1>     ;          DL = 0 -> 8x8 (width: 1 byte per row)
                                <1>     ;          DL = 1 -> 8x16
                                <1>     ;          DL = 2 -> 16x16 (width: 2 bytes)
                                <1>     ;          DL = 3 -> 16x32
                                <1>     ;          DL = 4 -> 24x24 (width: 3 bytes)
                                <1>     ;          DL = 5 -> 24x48
                                <1>     ;          DL = 6 -> 32x32 (width: 4 bytes)
                                <1>     ;          DL = 7 -> 32x64
                                <1>     ;          DL > 7 -> invalid (unused)
                                <1>     ;       EDI = user's font buffer address
                                <1>     ;          (NOTE: byte order is as row0,row1,row2..)
                                <1>     ;     ESI = start position (row, column) (*)
                                <1>     ;           (HW = row, SI = column)
                                <1>
0000E61C 89F0                   <1>     mov     eax, esi ; start position
0000E61E E856F2FFFF             <1>     call    calc_pixel_offset
0000E623 3B05[5E9D0100]         <1>     cmp     eax, [v_siz]
0000E629 736D                   <1>     jnb     short pix_op_chr_err ; out of display page!
0000E62B E826F2FFFF             <1>     call    pixels_to_byte_count
                                <1>     ; eax = font start offset
0000E630 0305[5A9D0100]         <1>     add     eax, [v_mem] ; LFB start address
0000E636 A3[629D0100]           <1>     mov     [v_str], eax ; font start address
                                <1>
0000E63B 890D[6A9D0100]         <1>     mov     [maskcolor], ecx ; save char's color
                                <1>
0000E641 8835[589D0100]         <1>     mov     [v_ops], dh
```

```
                                <1>
0000E647 81E6FFFF0000           <1>     and     esi, 0FFFFh
0000E64D 8935[729D0100]         <1>     mov     [buffer8], esi ; start column
                                <1>
0000E653 31DB                   <1>     xor     ebx, ebx ; 0
0000E655 31C0                   <1>     xor     eax, eax ; 15/02/2021
                                <1>
0000E657 F6C680                 <1>     test    dh, 80h
0000E65A 7577                   <1>     jnz     short pix_op_chr_u ; user font
                                <1>
0000E65C 80E63F                 <1>     and     dh, 3Fh ; clear bit 6, [UFONT] option bit
0000E65F 7409                   <1>     jz      short pix_op_chr_0
                                <1>
0000E661 80FE07                 <1>     cmp     dh, 7
0000E664 7732                   <1>     ja      short pix_op_chr_err
                                <1>                     ; invalid (undefined) option
0000E666 88F4                   <1>     mov     ah, dh
0000E668 D0EC                   <1>     shr     ah, 1
                                <1>     ; ah = 0 to 3, scale
                                <1>     ;jmp    short pix_op_chr_font_pixels
                                <1>
                                <1> pix_op_chr_font_pixels:
                                <1>     ; 05/02/2021
                                <1>     ; write scaled font to buffer
                                <1>
                                <1>     ; DL = ASCII code of character
                                <1>     ; AH = scale
                                <1>     ; EDI = buffer address (kernel)
                                <1>
                                <1> pix_op_chr_0:
0000E66A 88D3                   <1>     mov     bl, dl ; 15/02/2021
0000E66C 31C9                   <1>     xor     ecx, ecx
0000E66E B610                   <1>     mov     dh, 16
0000E670 F605[589D0100]01       <1>     test    byte [v_ops], 1 ; 8x8 font ?
0000E677 7428                   <1>     jz      short pix_op_chr_2 ; 8x16 font
0000E679 B608                   <1>     mov     dh, 8
0000E67B C1E303                 <1>     shl     ebx, 3 ; * 8
0000E67E F605[589D0100]40       <1>     test    byte [v_ops], 40h ; [ufont] option
0000E685 7412                   <1>     jz      short pix_op_chr_1  ; no
                                <1>     ; test 8x8 user font is ready flag
0000E687 F605[4E9D0100]01       <1>     test    byte [ufont], 1
0000E68E 7409                   <1>     jz      short pix_op_chr_1 ; no
0000E690 81C300500900           <1>     add     ebx, VGAFONT8USER
0000E696 EB2C                   <1>     jmp     short pix_op_chr_fpos_0
                                <1> pix_op_chr_err:
0000E698 C3                     <1>     retn
                                <1> pix_op_chr_1:
0000E699 81C3[104D0100]         <1>     add     ebx, vgafont8 ; system font (8x8)
0000E69F EB23                   <1>     jmp     short pix_op_chr_fpos_0
                                <1> pix_op_chr_2:
0000E6A1 C1E304                 <1>     shl     ebx, 4 ; * 16
0000E6A4 F605[589D0100]40       <1>     test    byte [v_ops], 40h ; [ufont] option
0000E6AB 7411                   <1>     jz      short pix_op_chr_3  ; no
                                <1>     ; test 8x16 user font is ready flag
0000E6AD F605[4E9D0100]02       <1>     test    byte [ufont], 2
0000E6B4 7408                   <1>     jz      short pix_op_chr_3 ; no
0000E6B6 81C300400900           <1>     add     ebx, VGAFONT16USER
0000E6BC EB06                   <1>     jmp     short pix_op_chr_fpos_0
                                <1> pix_op_chr_3:
0000E6BE 81C3[10630100]         <1>     add     ebx, vgafont16 ; system font (8x16)
                                <1> pix_op_chr_fpos_0:
0000E6C4 20E4                   <1>     and     ah, ah
0000E6C6 754A                   <1>     jnz     short pix_op_chr_fpos_1 ; scale > 1
                                <1>     ; no scale (scale = 1)
0000E6C8 89DE                   <1>     mov     esi, ebx ; 15/02/2021
0000E6CA 88F1                   <1>     mov     cl, dh ; rows/height (16 or 8)
0000E6CC B608                   <1>     mov     dh, 8  ; columns/width
0000E6CE E9CB000000             <1>     jmp     pix_op_chr_f2p
                                <1> pix_op_chr_u:
                                <1>     ; write user defined font
0000E6D3 80FE80                 <1>     cmp     dh, 80h
0000E6D6 75C0                   <1>     jne     short pix_op_chr_err
0000E6D8 80FA07                 <1>     cmp     dl, 7
0000E6DB 77BB                   <1>     ja      short pix_op_chr_err
                                <1>
                                <1>     ; 16/02/2021
0000E6DD 89FE                   <1>     mov     esi, edi ; user's font buffer
                                <1>
                                <1>     ;xor    eax, eax
                                <1>     ; eax = 0 ; 15/02/2021
0000E6DF 88D4                   <1>     mov     ah, dl
0000E6E1 D0EC                   <1>     shr     ah, 1
0000E6E3 FEC4                   <1>     inc     ah
                                <1>     ; ah =  1 to 4
0000E6E5 88E0                   <1>     mov     al, ah
0000E6E7 C0E003                 <1>     shl     al, 3 ; * 8
                                <1>     ; al = 8,16,24,32
0000E6EA 88C3                   <1>     mov     bl, al
0000E6EC 88C7                   <1>     mov     bh, al
0000E6EE F6E4                   <1>     mul     ah
                                <1>     ; ax = 8,32,72,128 bytes
0000E6F0 F6C201                 <1>     test    dl, 1
0000E6F3 7404                   <1>     jz      short pix_op_chr_u_0
                                <1>     ;shl    ax, 1 ; *2
                                <1>     ; 23/07/2022
0000E6F5 D1E0                   <1>     shl     eax, 1
                                <1>     ; ax = 16,32,144,256 bytes
0000E6F7 D0E7                   <1>     shl     bh, 1
                                <1> pix_op_chr_u_0:
                                <1>     ; eax = byte count
0000E6F9 89C1                   <1>     mov     ecx, eax
0000E6FB BF00760900             <1>     mov     edi, VBE3SAVERESTOREBLOCK
                                <1>     ; esi = user buffer
0000E700 E82F240000             <1>     call    transfer_from_user_buffer
0000E705 7291                   <1>     jc      short pix_op_chr_err
                                <1>
0000E707 88F9                   <1>     mov     cl, bh ; rows/height
0000E709 88DE                   <1>     mov     dh, bl ; columns (width)
0000E70B 89FE                   <1>     mov     esi, edi ; VBE3SAVERESTOREBLOCK
0000E70D E98C000000             <1>     jmp     pix_op_chr_f2p
                                <1>
                                <1> pix_op_chr_fpos_1:
                                <1>     ; 18/02/2021
                                <1>     ; scale > 1
0000E712 88F5                   <1>     mov     ch, dh ; 16 or 8
0000E714 BF00760900             <1>     mov     edi, VBE3SAVERESTOREBLOCK
0000E719 89FE                   <1>     mov     esi, edi
0000E71B FECC                   <1>     dec     ah
0000E71D 7522                   <1>     jnz     short pix_op_chr_fpos_5 ; scale > 2
                                <1>     ; scale = 2
                                <1> pix_op_chr_fpos_2:
0000E71F B108                   <1>     mov     cl, 8
0000E721 8A13                   <1>     mov     dl, [ebx]
                                <1> pix_op_chr_fpos_3:
```

```
7527                                <1>     ;shl    ax, 2
7528                                <1>     ; 23/07/2022
7529 0000E723 C1E002                <1>     shl     eax, 2
7530 0000E726 D0E2                  <1>     shl     dl, 1
7531 0000E728 7302                  <1>     jnc     short pix_op_chr_fpos_4
7532 0000E72A 0C03                  <1>     or      al, 3
7533                                <1> pix_op_chr_fpos_4:
7534 0000E72C FEC9                  <1>     dec     cl
7535 0000E72E 75F3                  <1>     jnz     short pix_op_chr_fpos_3
7536 0000E730 66AB                  <1>     stosw
7537                                <1>     ; 18/02/2021
7538 0000E732 66AB                  <1>     stosw
7539 0000E734 43                    <1>     inc     ebx
7540 0000E735 FECD                  <1>     dec     ch
7541 0000E737 75E6                  <1>     jnz     short pix_op_chr_fpos_2
7542                                <1>     ; scale = 2
7543 0000E739 88F1                  <1>     mov     cl, dh ; 16 or 8 (height/rows)
7544 0000E73B D0E1                  <1>     shl     cl, 1  ; 32 or 16 rows
7545 0000E73D B610                  <1>     mov     dh, 16 ; columns (width)
7546 0000E73F EB5D                  <1>     jmp     short pix_op_chr_f2p
7547                                <1> pix_op_chr_fpos_5:
7548 0000E741 FECC                  <1>     dec     ah
7549 0000E743 7538                  <1>     jnz     short pix_op_chr_fpos_9 ; scale = 4
7550                                <1>     ; scale = 3
7551                                <1> pix_op_chr_fpos_6:
7552 0000E745 B108                  <1>     mov     cl, 8
7553 0000E747 8A13                  <1>     mov     dl, [ebx]
7554                                <1> pix_op_chr_fpos_7:
7555 0000E749 C1E003                <1>     shl     eax, 3
7556 0000E74C D0E2                  <1>     shl     dl, 1 ; 18/02/2021
7557 0000E74E 7302                  <1>     jnc     short pix_op_chr_fpos_8
7558 0000E750 0C07                  <1>     or      al, 7
7559                                <1> pix_op_chr_fpos_8:
7560 0000E752 FEC9                  <1>     dec     cl
7561 0000E754 75F3                  <1>     jnz     short pix_op_chr_fpos_7
7562 0000E756 66AB                  <1>     stosw
7563                                <1>     ; 18/02/2021
7564 0000E758 C1C810                <1>     ror     eax, 16
7565 0000E75B AA                    <1>     stosb
7566 0000E75C C1C010                <1>     rol     eax, 16
7567 0000E75F 66AB                  <1>     stosw
7568 0000E761 C1C810                <1>     ror     eax, 16
7569 0000E764 AA                    <1>     stosb
7570 0000E765 C1C010                <1>     rol     eax, 16
7571 0000E768 66AB                  <1>     stosw
7572 0000E76A C1E810                <1>     shr     eax, 16 ; 27/02/2021
7573 0000E76D AA                    <1>     stosb
7574 0000E76E 43                    <1>     inc     ebx
7575                                <1>     ; 18/02/2021
7576 0000E76F FECD                  <1>     dec     ch
7577 0000E771 75D2                  <1>     jnz     short pix_op_chr_fpos_6
7578                                <1>     ; scale = 3
7579 0000E773 88F1                  <1>     mov     cl, dh ; 16 or 8 (height/rows)
7580 0000E775 D0E1                  <1>     shl     cl, 1
7581 0000E777 00F1                  <1>     add     cl, dh ; 48 or 24 rows
7582 0000E779 B618                  <1>     mov     dh, 24 ; columns (width)
7583 0000E77B EB21                  <1>     jmp     short pix_op_chr_f2p
7584                                <1>
7585                                <1> pix_op_chr_fpos_9:
7586                                <1>     ; scale = 4
7587 0000E77D B108                  <1>     mov     cl, 8
7588 0000E77F 8A13                  <1>     mov     dl, [ebx]
7589                                <1> pix_op_chr_fpos_10:
7590                                <1>     ; 18/02/2021
7591 0000E781 C1E004                <1>     shl     eax, 4
7592 0000E784 D0E2                  <1>     shl     dl, 1 ; 18/02/2021
7593 0000E786 7302                  <1>     jnc     short pix_op_chr_fpos_11
7594 0000E788 0C0F                  <1>     or      al, 0Fh ; or al, 15
7595                                <1> pix_op_chr_fpos_11:
7596 0000E78A FEC9                  <1>     dec     cl
7597 0000E78C 75F3                  <1>     jnz     short pix_op_chr_fpos_10
7598 0000E78E AB                    <1>     stosd
7599                                <1>     ; 18/02/2021
7600 0000E78F AB                    <1>     stosd
7601 0000E790 AB                    <1>     stosd
7602 0000E791 AB                    <1>     stosd
7603 0000E792 43                    <1>     inc     ebx
7604 0000E793 FECD                  <1>     dec     ch
7605 0000E795 75E6                  <1>     jnz     short pix_op_chr_fpos_9
7606                                <1>     ; scale = 4
7607 0000E797 88F1                  <1>     mov     cl, dh ; 16 or 8 (height/rows)
7608 0000E799 C0E102                <1>     shl     cl, 2 ; 64 or 32 rows
7609 0000E79C B620                  <1>     mov     dh, 32 ; columns (width)
7610                                <1>     ;jmp    short pix_op_chr_f2p
7611                                <1>
7612                                <1> pix_op_chr_f2p:
7613                                <1>     ; write font pixels
7614 0000E79E 8B3D[629D0100]        <1>     mov     edi, [v_str]
7615                                <1>     ; 15/02/2021
7616                                <1> pix_op_chr_f2p_next:
7617 0000E7A4 80FE08                <1>     cmp     dh, 8
7618 0000E7A7 7706                  <1>     ja      short pix_op_chr_f2p_24
7619                                <1> pix_op_chr_f2p_8:
7620 0000E7A9 AC                    <1>     lodsb
7621 0000E7AA C1E018                <1>     shl     eax, 24 ; 15/02/2021
7622 0000E7AD EB16                  <1>     jmp     short pix_op_chr_f2p_0
7623                                <1> pix_op_chr_f2p_24:
7624 0000E7AF 80FE18                <1>     cmp     dh, 24
7625 0000E7B2 7710                  <1>     ja      short pix_op_chr_f2p_32
7626 0000E7B4 7207                  <1>     jb      short pix_op_chr_f2p_16
7627                                <1>     ; 27/02/2021
7628                                <1>     ;mov    eax, [esi]
7629 0000E7B6 AD                    <1>     lodsd
7630 0000E7B7 C1E008                <1>     shl     eax, 8
7631                                <1>     ;add    esi, 3
7632 0000E7BA 4E                    <1>     dec     esi
7633 0000E7BB EB08                  <1>     jmp     short pix_op_chr_f2p_0
7634                                <1> pix_op_chr_f2p_16:
7635 0000E7BD 66AD                  <1>     lodsw
7636 0000E7BF C1E010                <1>     shl     eax, 16 ; 15/02/2021
7637 0000E7C2 EB01                  <1>     jmp     short pix_op_chr_f2p_0
7638                                <1> pix_op_chr_f2p_32:
7639 0000E7C4 AD                    <1>     lodsd
7640                                <1> pix_op_chr_f2p_0:
7641                                <1>     ; EAX = font row (8,16,24,32 pixels)
7642                                <1>     ;       (bits are shifted to left)
7643                                <1>     ; CL = rows
7644                                <1>     ; DH = bits per row (8,16,24,32)
7645 0000E7C5 8B1D[729D0100]        <1>     mov     ebx, [buffer8] ; start column
7646 0000E7CB 57                    <1>     push    edi ; *
7647 0000E7CC 52                    <1>     push    edx ; **
7648                                <1> pix_op_chr_f2p_1:
7649 0000E7CD E82E000000            <1>     call    pix_op_chr_w_pixel
7650                                <1> pix_op_chr_f2p_2:
```

```
7651 0000E7D2 663B1D[569D0100]    <1>     cmp     bx, [v_width] ; current column
7652 0000E7D9 7304                <1>     jnb     short pix_op_chr_f2p_3
7653 0000E7DB FECE                <1>     dec     dh
7654 0000E7DD 75EE                <1>     jnz     short pix_op_chr_f2p_1 ; next bit
7655                              <1> pix_op_chr_f2p_3:
7656                              <1>     ;mov    ebx, [buffer8]
7657 0000E7DF 5A                  <1>     pop     edx ; **
7658 0000E7E0 58                  <1>     pop     eax ; *
7659 0000E7E1 3B3D[669D0100]      <1>     cmp     edi, [v_end]
7660 0000E7E7 7316                <1>     jnb     short pix_op_chr_f2p_4
7661 0000E7E9 FEC9                <1>     dec     cl
7662 0000E7EB 7412                <1>     jz      short pix_op_chr_f2p_4
7663                              <1>     ; 27/02/2021
7664 0000E7ED 89C7                <1>     mov     edi, eax
7665 0000E7EF 0FB705[569D0100]    <1>     movzx   eax, word [v_width]
7666 0000E7F6 E85BF0FFFF          <1>     call    pixels_to_byte_count
7667 0000E7FB 01C7                <1>     add     edi, eax ; next position
7668 0000E7FD EBA5                <1>     jmp     short pix_op_chr_f2p_next
7669                              <1> pix_op_chr_f2p_4:
7670 0000E7FF C3                  <1>     retn
7671                              <1>
7672                              <1> pix_op_chr_w_pixel:
7673                              <1>     ; 15/02/2021
7674 0000E800 89C5                <1>     mov     ebp, eax
7675 0000E802 A1[6A9D0100]        <1>     mov     eax, [maskcolor]
7676 0000E807 803D[599D0100]08    <1>     cmp     byte [v_bpp], 8 ; 8bpp
7677 0000E80E 7711                <1>     ja      short pix_op_chr_wp_2
7678                              <1>     ; 256 colors (1 byte per pixel)
7679 0000E810 D1E5                <1>     shl     ebp, 1
7680 0000E812 7302                <1>     jnc     short pix_op_chr_wp_0
7681 0000E814 8807                <1>     mov     [edi], al
7682                              <1> pix_op_chr_wp_0:
7683 0000E816 47                  <1>     inc     edi
7684 0000E817 FF05[348E0100]      <1>     inc     dword [u.r0] ; +1
7685                              <1> pix_op_chr_wp_1:
7686 0000E81D 43                  <1>     inc     ebx
7687 0000E81E 89E8                <1>     mov     eax, ebp
7688 0000E820 C3                  <1>     retn
7689                              <1> pix_op_chr_wp_2:
7690 0000E821 803D[599D0100]18    <1>     cmp     byte [v_bpp], 24 ; 24bpp
7691 0000E828 772E                <1>     ja      short pix_op_chr_wp_6 ; 32bpp
7692 0000E82A 721C                <1>     jb      short pix_op_chr_wp_4 ; 16bpp
7693                              <1>     ; 24 bit true colors
7694                              <1>     ; * 3
7695 0000E82C D1E5                <1>     shl     ebp, 1
7696 0000E82E 7309                <1>     jnc     short pix_op_chr_wp_3
7697 0000E830 668907              <1>     mov     [edi], ax
7698 0000E833 C1E810              <1>     shr     eax, 16
7699 0000E836 884702              <1>     mov     [edi+2], al
7700                              <1> pix_op_chr_wp_3:
7701 0000E839 B803000000          <1>     mov     eax, 3 ; 27/02/2021
7702 0000E83E 01C7                <1>     add     edi, eax  ; add edi, 3
7703 0000E840 0105[348E0100]      <1>     add     [u.r0], eax ; +3
7704                              <1>
7705 0000E846 EBD5                <1>     jmp     short pix_op_chr_wp_1
7706                              <1>
7707                              <1> pix_op_chr_wp_4:
7708                              <1>     ; 16 bit (65536) colors
7709 0000E848 D1E5                <1>     shl     ebp, 1
7710 0000E84A 7303                <1>     jnc     short pix_op_chr_wp_5
7711 0000E84C 668907              <1>     mov     [edi], ax
7712                              <1> pix_op_chr_wp_5:
7713 0000E84F 47                  <1>     inc     edi
7714 0000E850 FF05[348E0100]      <1>     inc     dword [u.r0] ; +1
7715 0000E856 EBBE                <1>     jmp     short pix_op_chr_wp_0
7716                              <1>
7717                              <1> pix_op_chr_wp_6:
7718                              <1>     ; 32 bit true colors
7719 0000E858 D1E5                <1>     shl     ebp, 1
7720 0000E85A 7302                <1>     jnc     short pix_op_chr_wp_7
7721 0000E85C 8907                <1>     mov     [edi], eax
7722                              <1> pix_op_chr_wp_7:
7723 0000E85E 31C0                <1>     xor     eax, eax
7724 0000E860 B004                <1>     mov     al, 4
7725 0000E862 01C7                <1>     add     edi, eax ; add edi, 4
7726 0000E864 0105[348E0100]      <1>     add     [u.r0], eax ; +4
7727 0000E86A EBB1                <1>     jmp     short pix_op_chr_wp_1
7728                              <1>
7729                              <1> m_pix_op_cpy:
7730                              <1>     ; 26/02/2021
7731                              <1>     ; 06/02/2021
7732                              <1>     ; MASKED COPY PIXELS (full screen)
7733                              <1>     ;
7734                              <1>     ; jump from pix_op_cpy
7735                              <1>     ;
7736                              <1>     ; INPUT:
7737                              <1>     ;   ecx = transfer count (bytes)
7738                              <1>     ;   edi = [v_mem] = start address of LFB
7739                              <1>     ;   esi = user's buffer address (virtual)
7740                              <1>     ;
7741                              <1>     ; OUTPUT:
7742                              <1>     ;   [u.r0] will be > 0 if succesful
7743                              <1>
7744                              <1>     ; Full screen masked copy
7745                              <1>
7746                              <1>     ; Modified regs: eax, ebx, edx, esi, edi, ecx
7747                              <1>
7748                              <1> m_pix_op_cpy_0:
7749                              <1>     ;push   ebx ; *** ; 26/02/2021
7750 0000E86C 57                  <1>     push    edi ; **
7751 0000E86D 51                  <1>     push    ecx ; *
7752 0000E86E 81F9F8070000        <1>     cmp     ecx, 2040 ; (3*680) ; 26/02/2021
7753 0000E874 7605                <1>     jna     short m_pix_op_cpy_1
7754 0000E876 B9F8070000          <1>     mov     ecx, 2040
7755                              <1> m_pix_op_cpy_1:
7756 0000E87B BF00760900          <1>     mov     edi, VBE3SAVERESTOREBLOCK ; temporary buff
7757 0000E880 E8AF220000          <1>     call    transfer_from_user_buffer
7758 0000E885 726C                <1>     jc      short m_pix_op_cpy_3
7759 0000E887 01CE                <1>     add     esi, ecx
7760 0000E889 89F5                <1>     mov     ebp, esi  ; save user's buffer address
7761 0000E88B 89FE                <1>     mov     esi, edi
7762 0000E88D 89CB                <1>     mov     ebx, ecx
7763 0000E88F 59                  <1>     pop     ecx ; *
7764 0000E890 29D9                <1>     sub     ecx, ebx
7765 0000E892 5F                  <1>     pop     edi ; **
7766 0000E893 31D2                <1>     xor     edx, edx ; 26/02/2021
7767 0000E895 803D[599D0100]08    <1>     cmp     byte [v_bpp], 8
7768 0000E89C 7435                <1>     je      short m_pix_op_cpy_1_8
7769 0000E89E 803D[599D0100]18    <1>     cmp     byte [v_bpp], 24
7770 0000E8A5 776D                <1>     ja      short m_pix_op_cpy_1_32
7771 0000E8A7 724D                <1>     jb      short m_pix_op_cpy_1_16
7772                              <1> m_pix_op_cpy_1_24:
7773                              <1>     ; 24 bit masked copy
7774                              <1>     ;mov    edx, 3
```

```
7775 0000E8A9 B203              <1>     mov     dl, 3 ; 26/02/2021
7776                            <1> m_pix_op_cpy_1_24_0:
7777 0000E8AB 66AD              <1>     lodsw
7778 0000E8AD C1E010            <1>     shl     eax, 16
7779 0000E8B0 AC                <1>     lodsb
7780 0000E8B1 C1C010            <1>     rol     eax, 16
7781 0000E8B4 3B05[6A9D0100]    <1>     cmp     eax, [maskcolor]
7782 0000E8BA 740F              <1>     je      short m_pix_op_cpy_1_24_1 ; exclude
7783 0000E8BC 668907            <1>     mov     [edi], ax
7784 0000E8BF C1E810            <1>     shr     eax, 16
7785 0000E8C2 884702            <1>     mov     [edi+2], al
7786 0000E8C5 0115[348E0100]    <1>     add     [u.r0], edx ; +3
7787                            <1> m_pix_op_cpy_1_24_1:
7788 0000E8CB 01D7              <1>     add     edi, edx ; +3
7789 0000E8CD 29D3              <1>     sub     ebx, edx ; sub ebx, 3
7790 0000E8CF 77DA              <1>     ja      short m_pix_op_cpy_1_24_0
7791 0000E8D1 EB15              <1>     jmp     short m_pix_op_cpy_2
7792                            <1>
7793                            <1> m_pix_op_cpy_1_8:
7794                            <1>     ; 8 bit masked copy
7795 0000E8D3 AC                <1>     lodsb
7796 0000E8D4 3A05[6A9D0100]    <1>     cmp     al, [maskcolor]
7797 0000E8DA 7408              <1>     je      short m_pix_op_cpy_1_8_1 ; exclude
7798 0000E8DC 8807              <1>     mov     [edi], al
7799 0000E8DE FF05[348E0100]    <1>     inc     dword [u.r0] ; +1
7800                            <1> m_pix_op_cpy_1_8_1:
7801 0000E8E4 47                <1>     inc     edi ; +1
7802 0000E8E5 4B                <1>     dec     ebx
7803 0000E8E6 75EB              <1>     jnz     short m_pix_op_cpy_1_8
7804                            <1> m_pix_op_cpy_2:
7805 0000E8E8 89EE              <1>     mov     esi, ebp ; restore user's buffer addr
7806 0000E8EA 09C9              <1>     or      ecx, ecx
7807                            <1>     ; 26/02/2021
7808 0000E8EC 7407              <1>     jz      short m_pix_of_cpy_4
7809 0000E8EE E979FFFFFF        <1>     jmp     m_pix_op_cpy_0
7810                            <1> m_pix_op_cpy_3:
7811 0000E8F3 59                <1>     pop     ecx ; *
7812 0000E8F4 5F                <1>     pop     edi ; **
7813                            <1> m_pix_of_cpy_4:
7814                            <1>     ;pop    ebx ; *** ; 26/02/2021
7815 0000E8F5 C3                <1>     retn
7816                            <1>
7817                            <1> m_pix_op_cpy_1_16:
7818                            <1>     ; 16 bit masked copy
7819                            <1>     ;mov    edx, 2
7820 0000E8F6 B202              <1>     mov     dl, 2 ; 26/02/2021
7821                            <1> m_pix_op_cpy_1_16_0:
7822 0000E8F8 66AD              <1>     lodsw
7823 0000E8FA 663B05[6A9D0100]  <1>     cmp     ax, [maskcolor]
7824 0000E901 7409              <1>     je      short m_pix_op_cpy_1_16_1 ; exclude
7825 0000E903 668907            <1>     mov     [edi], ax
7826 0000E906 0115[348E0100]    <1>     add     [u.r0], edx ; +2
7827                            <1> m_pix_op_cpy_1_16_1:
7828 0000E90C 01D7              <1>     add     edi, edx ; +2
7829 0000E90E 29D3              <1>     sub     ebx, edx ; sub ebx, 2
7830 0000E910 77E6              <1>     ja      short m_pix_op_cpy_1_16_0
7831 0000E912 EBD4              <1>     jmp     short m_pix_op_cpy_2
7832                            <1>
7833                            <1> m_pix_op_cpy_1_32:
7834                            <1>     ; 32 bit masked copy
7835                            <1>     ;mov    edx, 4
7836 0000E914 B204              <1>     mov     dl, 4 ; 26/02/2021
7837                            <1> m_pix_op_cpy_1_32_0:
7838 0000E916 AD                <1>     lodsd
7839 0000E917 3B05[6A9D0100]    <1>     cmp     eax, [maskcolor]
7840 0000E91D 7408              <1>     je      short m_pix_op_cpy_1_32_1 ; exclude
7841 0000E91F 8907              <1>     mov     [edi], eax
7842 0000E921 0115[348E0100]    <1>     add     [u.r0], edx ; +4
7843                            <1> m_pix_op_cpy_1_32_1:
7844 0000E927 01D7              <1>     add     edi, edx ; +4
7845 0000E929 29D3              <1>     sub     ebx, edx ; sub ebx, 4
7846 0000E92B 77E9              <1>     ja      short m_pix_op_cpy_1_32_0
7847 0000E92D EBB9              <1>     jmp     short m_pix_op_cpy_2
7848                            <1>
7849                            <1> m_pix_op_cpy_w:
7850                            <1>     ; 26/02/2021
7851                            <1>     ; 06/02/2021
7852                            <1>     ; MASKED COPY PIXELS (window)
7853                            <1>     ;
7854                            <1>     ; jump from pix_op_cpy_w
7855                            <1>     ;
7856                            <1>     ; INPUT:
7857                            <1>     ;   ecx = bytes per row (to be applied)
7858                            <1>     ;   edx = screen width in bytes
7859                            <1>     ;   ebx = row count
7860                            <1>     ;   esi = user's buffer address
7861                            <1>     ;   [v_str] = window start address
7862                            <1>     ;
7863                            <1>     ; OUTPUT:
7864                            <1>     ;   [u.r0] will be > 0 if succesful
7865                            <1>
7866                            <1>     ; Window masked copy
7867                            <1>
7868                            <1> m_pix_op_cpy_w_0:
7869 0000E92F 8B3D[629D0100]    <1>     mov     edi, [v_str]
7870                            <1> m_pix_op_cpy_w_1:
7871 0000E935 57                <1>     push    edi
7872 0000E936 56                <1>     push    esi
7873 0000E937 53                <1>     push    ebx
7874 0000E938 52                <1>     push    edx
7875 0000E939 51                <1>     push    ecx
7876 0000E93A E82DFFFFFF        <1>     call    m_pix_op_cpy ; 26/02/2021
7877 0000E93F 59                <1>     pop     ecx
7878 0000E940 5A                <1>     pop     edx
7879 0000E941 5B                <1>     pop     ebx
7880 0000E942 5E                <1>     pop     esi
7881 0000E943 5F                <1>     pop     edi
7882 0000E944 7209              <1>     jc      short m_pix_op_cpy_w_2
7883 0000E946 4B                <1>     dec     ebx
7884 0000E947 7406              <1>     jz      short m_pix_op_cpy_w_2 ; ok.
7885                            <1>     ; next row
7886 0000E949 01CE              <1>     add     esi, ecx ; next row in user's buffer
7887 0000E94B 01D7              <1>     add     edi, edx ; next row of window (system)
7888 0000E94D EBE6              <1>     jmp     short m_pix_op_cpy_w_1
7889                            <1> m_pix_op_cpy_w_2:
7890 0000E94F C3                <1>     retn
7891                            <1>
7892                            <1> m_pix_op_new:
7893                            <1>     ; 06/02/2021
7894                            <1>     ; CHANGE COLOR (MASKED, full screen)
7895                            <1>     ;
7896                            <1>     ; jump from pix_op_new
7897                            <1>     ;
7898                            <1>     ; INPUT:
```

```
                                <1>     ;   eax = color (AL, AX, EAX)
                                <1>     ;   ecx = [v_siz] ; display page pixel count
                                <1>     ;   esi = edi = [v_mem] ; LFB start address
                                <1>     ;
                                <1>     ;   [maskcolor] = mask color (to be excluded)
                                <1>     ;
                                <1>     ; OUTPUT:
                                <1>     ;       [u.r0] will be > 0 if succesful
                                <1>
                                <1>     ; Full screen
                                <1> m_pix_op_new_0:
0000E950 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000E957 7717                   <1>     ja      short m_pix_op_new_1
                                <1>     ; 256 colors (8bpp)
                                <1>     ;jmp    short m_pix_op_new_8
                                <1> m_pix_op_new_8:
                                <1>     ; 8 bit colors (256 colors)
0000E959 88C2                   <1>     mov     dl, al ; new color
                                <1> m_pix_op_new_8_0:
0000E95B AC                     <1>     lodsb
0000E95C 3A05[6A9D0100]         <1>     cmp     al, [maskcolor]
0000E962 7408                   <1>     je      short m_pix_op_new_8_1 ; exclude
0000E964 8817                   <1>     mov     [edi], dl
0000E966 FF05[348E0100]         <1>     inc     dword [u.r0]
                                <1> m_pix_op_new_8_1:
0000E96C 47                     <1>     inc     edi
0000E96D E2EC                   <1>     loop    m_pix_op_new_8_0
0000E96F C3                     <1>     retn
                                <1> m_pix_op_new_1:
0000E970 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000E977 774B                   <1>     ja      short m_pix_op_new_3 ; 32bpp
0000E979 722C                   <1>     jb      short m_pix_op_new_2 ; 16bpp
                                <1>     ; 24 bit true colors
                                <1>     ;jmp    short m_pix_op_new_24
                                <1> m_pix_op_new_24:
                                <1>     ; 24 bit true colors
0000E97B 89C2                   <1>     mov     edx, eax ; new color
                                <1> m_pix_op_new_24_0:
0000E97D 66AD                   <1>     lodsw
0000E97F C1E010                 <1>     shl     eax, 16
0000E982 AC                     <1>     lodsb
0000E983 C1C010                 <1>     rol     eax, 16
0000E986 3B05[6A9D0100]         <1>     cmp     eax, [maskcolor]
0000E98C 7413                   <1>     je      short m_pix_op_new_24_1 ; exclude
0000E98E 668917                 <1>     mov     [edi], dx
0000E991 C1CA10                 <1>     ror     edx, 16
0000E994 885702                 <1>     mov     [edi+2], dl
0000E997 C1C210                 <1>     rol     edx, 16
0000E99A 8305[348E0100]03       <1>     add     dword [u.r0], 3
                                <1> m_pix_op_new_24_1:
0000E9A1 83C703                 <1>     add     edi, 3
0000E9A4 E2D7                   <1>     loop    m_pix_op_new_24_0
0000E9A6 C3                     <1>     retn
                                <1>     ; 65536 colors (16bpp)
                                <1> m_pix_op_new_2:
                                <1>     ;jmp    short m_pix_op_new_16
                                <1> m_pix_op_new_16:
                                <1>     ; 16 bit colors (65536 colors)
0000E9A7 89C2                   <1>     mov     edx, eax ; new color
                                <1> m_pix_op_new_16_0:
0000E9A9 66AD                   <1>     lodsw
0000E9AB 663B05[6A9D0100]       <1>     cmp     ax, [maskcolor]
0000E9B2 740A                   <1>     je      short m_pix_op_new_16_1 ; exclude
0000E9B4 668917                 <1>     mov     [edi], dx
0000E9B7 8305[348E0100]02       <1>     add     dword [u.r0], 2
                                <1> m_pix_op_new_16_1:
0000E9BE 83C702                 <1>     add     edi, 2
0000E9C1 E2E6                   <1>     loop    m_pix_op_new_16_0
0000E9C3 C3                     <1>     retn
                                <1> m_pix_op_new_3:
                                <1>     ; 32 bit true colors
                                <1>     ;jmp    short m_pix_op_new_32
                                <1> m_pix_op_new_32:
                                <1>     ; 32 bit true colors
0000E9C4 89C2                   <1>     mov     edx, eax ; new color
                                <1> m_pix_op_new_32_0:
0000E9C6 AD                     <1>     lodsd
0000E9C7 3B05[6A9D0100]         <1>     cmp     eax, [maskcolor]
0000E9CD 7409                   <1>     je      short m_pix_op_new_32_1 ; exclude
0000E9CF 8917                   <1>     mov     [edi], edx
0000E9D1 8305[348E0100]04       <1>     add     dword [u.r0], 4
                                <1> m_pix_op_new_32_1:
0000E9D8 83C704                 <1>     add     edi, 4
0000E9DB E2E9                   <1>     loop    m_pix_op_new_32_0
0000E9DD C3                     <1>     retn
                                <1>
                                <1> m_pix_op_new_w:
                                <1>     ; 06/02/2021
                                <1>     ; CHANGE COLOR (MASKED, window)
                                <1>     ;
                                <1>     ; jump from pix_op_new_w
                                <1>     ;
                                <1>     ; INPUT:
                                <1>     ;   ecx = bytes per row (to be applied)
                                <1>     ;   edx = screen width in bytes
                                <1>     ;   ebx = row count
                                <1>     ;   eax = color
                                <1>     ;
                                <1>     ;   [maskcolor] = mask color (to be excluded)
                                <1>     ;
                                <1>     ; OUTPUT:
                                <1>     ;       [u.r0] will be > 0 if succesful
                                <1>
                                <1>     ; Window
                                <1>     ;mov    edi, [v_str] ; LFB start address
                                <1>     ;mov    esi, edi
                                <1>
0000E9DE 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000E9E5 7707                   <1>     ja      short m_pix_op_new_w_1
                                <1>
                                <1>     ; 256 colors (8bpp)
0000E9E7 BD[59E90000]           <1>     mov     ebp, m_pix_op_new_8
0000E9EC EB1E                   <1>     jmp     short m_pix_op_new_w_x
                                <1>
                                <1> m_pix_op_new_w_1:
0000E9EE 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000E9F5 7710                   <1>     ja      short m_pix_op_new_w_3 ; 32bpp
0000E9F7 7207                   <1>     jb      short m_pix_op_new_w_2 ; 16bpp
                                <1>
                                <1>     ; 24 bit true colors
0000E9F9 BD[7BE90000]           <1>     mov     ebp, m_pix_op_new_24
0000E9FE EB0C                   <1>     jmp     short m_pix_op_new_w_x
                                <1>
                                <1>     ; 65536 colors (16bpp)
```

```
                                <1> m_pix_op_new_w_2:
0000EA00 BD[A7E90000]           <1>     mov     ebp, m_pix_op_new_16
0000EA05 EB05                   <1>     jmp     short m_pix_op_new_w_x
                                <1>
                                <1>     ; 32 bit true colors
                                <1> m_pix_op_new_w_3:
0000EA07 BD[C4E90000]           <1>     mov     ebp, m_pix_op_new_32
                                <1>     ;jmp    short m_pix_op_new_w_x
                                <1>
                                <1> m_pix_op_new_w_x:
                                <1> m_pix_op_add_w_x:
                                <1> m_pix_op_sub_w_x:
                                <1> m_pix_op_mix_w_x:
                                <1> m_pix_op_and_w_x:
                                <1> m_pix_op_orc_w_x:
                                <1> m_pix_op_xor_w_x:
                                <1> m_pix_op_not_w_x:
                                <1> m_pix_op_neg_w_x:
                                <1> m_pix_op_inc_w_x:
                                <1> m_pix_op_dec_w_x:
                                <1>     ; 27/02/2021
                                <1>     ; 26/02/2021
                                <1>     ; 06/02/2021
                                <1>     ; ecx = bytes per row (to be applied)
                                <1>     ; edx = windows (screen) width in bytes
                                <1>     ; ebx = row count
                                <1>     ; eax = color
                                <1>     ; ebp = pixel operation subroutine address
                                <1>     ; edi = esi = window start address
                                <1>
0000EA0C 8B3D[629D0100]         <1>     mov     edi, [v_str] ; LFB start address
0000EA12 89FE                   <1>     mov     esi, edi
                                <1> m_pix_op_w_x_next:
0000EA14 52                     <1>     push    edx
0000EA15 51                     <1>     push    ecx
0000EA16 56                     <1>     push    esi
0000EA17 57                     <1>     push    edi
0000EA18 50                     <1>     push    eax ; 26/02/2021
0000EA19 8B0D[6E9D0100]         <1>     mov     ecx, [pixcount] ; 27/02/2021
0000EA1F FFD5                   <1>     call    ebp ; call masked pixel-row operation
0000EA21 58                     <1>     pop     eax ; 26/02/2021
0000EA22 5F                     <1>     pop     edi
0000EA23 5E                     <1>     pop     esi
0000EA24 59                     <1>     pop     ecx
0000EA25 5A                     <1>     pop     edx
0000EA26 01D6                   <1>     add     esi, edx ; next row
0000EA28 01D7                   <1>     add     edi, edx ; next row
0000EA2A 4B                     <1>     dec     ebx
0000EA2B 75E7                   <1>     jnz     short m_pix_op_w_x_next
0000EA2D C3                     <1>     retn
                                <1>
                                <1> m_pix_op_add:
                                <1>     ; 06/02/2021
                                <1>     ; ADD COLOR (MASKED, full screen)
                                <1>     ;
                                <1>     ; jump from pix_op_add
                                <1>     ;
                                <1>     ; INPUT:
                                <1>     ;   eax = color (AL, AX, EAX)
                                <1>     ;   ecx = [v_siz] ; display page pixel count
                                <1>     ;   esi = edi = [v_mem] ; LFB start address
                                <1>     ;
                                <1>     ;   [maskcolor] = mask color (to be excluded)
                                <1>     ;
                                <1>     ; OUTPUT:
                                <1>     ;       [u.r0] will be > 0 if succesful
                                <1>
                                <1>     ; Full screen
                                <1> m_pix_op_add_0:
0000EA2E 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000EA35 771C                   <1>     ja      short m_pix_op_add_1
                                <1>     ; 256 colors (8bpp)
                                <1>     ;jmp    short m_pix_op_add_8
                                <1> m_pix_op_add_8:
                                <1>     ; 8 bit colors (256 colors)
0000EA37 88C2                   <1>     mov     dl, al ; new color
                                <1> m_pix_op_add_8_0:
0000EA39 AC                     <1>     lodsb
0000EA3A 3A05[6A9D0100]         <1>     cmp     al, [maskcolor]
0000EA40 740D                   <1>     je      short m_pix_op_add_8_1 ; exclude
0000EA42 FF05[348E0100]         <1>     inc     dword [u.r0] ; +1
0000EA48 0017                   <1>     add     [edi], dl
0000EA4A 7303                   <1>     jnc     short m_pix_op_add_8_1
0000EA4C C607FF                 <1>     mov     byte [edi], 0FFh
                                <1> m_pix_op_add_8_1:
0000EA4F 47                     <1>     inc     edi
0000EA50 E2E7                   <1>     loop    m_pix_op_add_8_0
0000EA52 C3                     <1>     retn
                                <1> m_pix_op_add_1:
0000EA53 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000EA5A 775E                   <1>     ja      short m_pix_op_add_3 ; 32bpp
0000EA5C 7238                   <1>     jb      short m_pix_op_add_2 ; 16bpp
                                <1>     ; 24 bit true colors
                                <1>     ;jmp    short m_pix_op_add_24
                                <1> m_pix_op_add_24:
                                <1>     ; 24 bit true colors
0000EA5E 89C2                   <1>     mov     edx, eax ; new color
0000EA60 81CA000000FF           <1>     or      edx, 0FF000000h
                                <1> m_pix_op_add_24_0:
0000EA66 66AD                   <1>     lodsw
0000EA68 C1E010                 <1>     shl     eax, 16
0000EA6B AC                     <1>     lodsb
0000EA6C C1C010                 <1>     rol     eax, 16
0000EA6F 3B05[6A9D0100]         <1>     cmp     eax, [maskcolor]
0000EA75 7419                   <1>     je      short m_pix_op_add_24_2 ; exclude
0000EA77 8305[348E0100]03       <1>     add     dword [u.r0], 3 ; +3
0000EA7E 01D0                   <1>     add     eax, edx
0000EA80 7305                   <1>     jnc     short m_pix_op_add_24_1
0000EA82 B8FFFFFF00             <1>     mov     eax, 0FFFFFFh
                                <1> m_pix_op_add_24_1:
0000EA87 668907                 <1>     mov     [edi], ax
0000EA8A C1E810                 <1>     shr     eax, 16
0000EA8D 884702                 <1>     mov     [edi+2], al
                                <1> m_pix_op_add_24_2:
0000EA90 83C703                 <1>     add     edi, 3 ; +3
0000EA93 E2D1                   <1>     loop    m_pix_op_add_24_0
0000EA95 C3                     <1>     retn
                                <1>     ; 65536 colors (16bpp)
                                <1> m_pix_op_add_2:
                                <1>     ;jmp    short m_pix_op_add_16
                                <1> m_pix_op_add_16:
                                <1>     ; 16 bit colors (65536 colors)
0000EA96 89C2                   <1>     mov     edx, eax ; new color
                                <1> m_pix_op_add_16_0:
```

```
8147 0000EA98 66AD                <1>     lodsw
8148 0000EA9A 663B05[6A9D0100]    <1>     cmp     ax, [maskcolor]
8149 0000EAA1 7411                <1>     je      short m_pix_op_add_16_1 ; exclude
8150 0000EAA3 8305[348E0100]02    <1>     add     dword [u.r0], 2 ; +2
8151 0000EAAA 660117              <1>     add     [edi], dx
8152 0000EAAD 7305                <1>     jnc     short m_pix_op_add_16_1
8153 0000EAAF 66C707FFFF          <1>     mov     word [edi], 0FFFFh
8154                              <1> m_pix_op_add_16_1:
8155 0000EAB4 83C702              <1>     add     edi, 2 ; +2
8156 0000EAB7 E2DF                <1>     loop    m_pix_op_add_16_0
8157 0000EAB9 C3                  <1>     retn
8158                              <1> m_pix_op_add_3:
8159                              <1>     ; 32 bit true colors
8160                              <1>     ;jmp    short m_pix_op_add_32
8161                              <1> m_pix_op_add_32:
8162                              <1>     ; 32 bit true colors
8163 0000EABA 89C2                <1>     mov     edx, eax ; new color
8164                              <1> m_pix_op_add_32_0:
8165 0000EABC AD                  <1>     lodsd
8166 0000EABD 3B05[6A9D0100]      <1>     cmp     eax, [maskcolor]
8167 0000EAC3 7411                <1>     je      short m_pix_op_add_32_1 ; exclude
8168 0000EAC5 8305[348E0100]04    <1>     add     dword [u.r0], 4 ; +4
8169 0000EACC 0117                <1>     add     [edi], edx
8170 0000EACE 7306                <1>     jnc     short m_pix_op_add_32_1
8171 0000EAD0 C707FFFFFFFF        <1>     mov     dword [edi], 0FFFFFFFFh
8172                              <1> m_pix_op_add_32_1:
8173 0000EAD6 83C704              <1>     add     edi, 4 ; +4
8174 0000EAD9 E2E1                <1>     loop    m_pix_op_add_32_0
8175 0000EADB C3                  <1>     retn
8176                              <1>
8177                              <1> m_pix_op_add_w:
8178                              <1>     ; 06/02/2021
8179                              <1>     ; ADD COLOR (MASKED, window)
8180                              <1>     ;
8181                              <1>     ; jump from pix_op_add_w
8182                              <1>     ;
8183                              <1>     ; INPUT:
8184                              <1>     ;   ecx = bytes per row (to be applied)
8185                              <1>     ;   edx = screen width in bytes
8186                              <1>     ;   ebx = row count
8187                              <1>     ;   eax = color
8188                              <1>     ;
8189                              <1>     ;   [maskcolor] = mask color (to be excluded)
8190                              <1>     ;
8191                              <1>     ; OUTPUT:
8192                              <1>     ;       [u.r0] will be > 0 if succesful
8193                              <1>
8194                              <1>     ; window
8195                              <1>     ;mov    edi, [v_str] ; LFB start address
8196                              <1>     ;mov    esi, edi
8197                              <1>
8198 0000EADC 803D[599D0100]08    <1>     cmp     byte [v_bpp], 8 ; 8bpp
8199 0000EAE3 7707                <1>     ja      short m_pix_op_add_w_1
8200                              <1>
8201                              <1>     ; 256 colors (8bpp)
8202 0000EAE5 BD[37EA0000]        <1>     mov     ebp, m_pix_op_add_8
8203 0000EAEA EB1E                <1>     jmp     short m_pix_op_add_w_4
8204                              <1>
8205                              <1> m_pix_op_add_w_1:
8206 0000EAEC 803D[599D0100]18    <1>     cmp     byte [v_bpp], 24 ; 24bpp
8207 0000EAF3 7710                <1>     ja      short m_pix_op_add_w_3 ; 32bpp
8208 0000EAF5 7207                <1>     jb      short m_pix_op_add_w_2 ; 16bpp
8209                              <1>
8210                              <1>     ; 24 bit true colors
8211 0000EAF7 BD[5EEA0000]        <1>     mov     ebp, m_pix_op_add_24
8212 0000EAFC EB0C                <1>     jmp     short m_pix_op_add_w_4
8213                              <1>
8214                              <1>     ; 65536 colors (16bpp)
8215                              <1> m_pix_op_add_w_2:
8216 0000EAFE BD[96EA0000]        <1>     mov     ebp, m_pix_op_add_16
8217 0000EB03 EB05                <1>     jmp     short m_pix_op_add_w_4
8218                              <1>
8219                              <1>     ; 32 bit true colors
8220                              <1> m_pix_op_add_w_3:
8221 0000EB05 BD[BAEA0000]        <1>     mov     ebp, m_pix_op_add_32
8222                              <1> m_pix_op_add_w_4:
8223 0000EB0A E9FDFEFFFF          <1>     jmp     m_pix_op_add_w_x
8224                              <1>
8225                              <1> m_pix_op_sub:
8226                              <1>     ; 02/03/2021
8227                              <1>     ; 06/02/2021
8228                              <1>     ; SUBTRACT COLOR (MASKED, full screen)
8229                              <1>     ;
8230                              <1>     ; jump from pix_op_sub
8231                              <1>     ;
8232                              <1>     ; INPUT:
8233                              <1>     ;   eax = color (AL, AX, EAX)
8234                              <1>     ;   ecx = [v_siz] ; display page pixel count
8235                              <1>     ;   esi = edi = [v_mem] ; LFB start address
8236                              <1>     ;
8237                              <1>     ;   [maskcolor] = mask color (to be excluded)
8238                              <1>     ;
8239                              <1>     ; OUTPUT:
8240                              <1>     ;       [u.r0] will be > 0 if succesful
8241                              <1>
8242                              <1>     ; Full screen
8243                              <1> m_pix_op_sub_0:
8244 0000EB0F 803D[599D0100]08    <1>     cmp     byte [v_bpp], 8 ; 8bpp
8245 0000EB16 771C                <1>     ja      short m_pix_op_sub_1
8246                              <1>     ; 256 colors (8bpp)
8247                              <1>     ;jmp    short m_pix_op_sub_8
8248                              <1> m_pix_op_sub_8:
8249                              <1>     ; 8 bit colors (256 colors)
8250 0000EB18 88C2                <1>     mov     dl, al ; new color
8251                              <1> m_pix_op_sub_8_0:
8252 0000EB1A AC                  <1>     lodsb
8253 0000EB1B 3A05[6A9D0100]      <1>     cmp     al, [maskcolor]
8254 0000EB21 740D                <1>     je      short m_pix_op_sub_8_1 ; exclude
8255 0000EB23 FF05[348E0100]      <1>     inc     dword [u.r0] ; +1
8256 0000EB29 2817                <1>     sub     [edi], dl
8257 0000EB2B 7303                <1>     jnb     short m_pix_op_sub_8_1
8258 0000EB2D C60700              <1>     mov     byte [edi], 0
8259                              <1> m_pix_op_sub_8_1:
8260 0000EB30 47                  <1>     inc     edi
8261 0000EB31 E2E7                <1>     loop    m_pix_op_sub_8_0
8262 0000EB33 C3                  <1>     retn
8263                              <1> m_pix_op_sub_1:
8264 0000EB34 803D[599D0100]18    <1>     cmp     byte [v_bpp], 24 ; 24bpp
8265 0000EB3B 7755                <1>     ja      short m_pix_op_sub_3 ; 32bpp
8266 0000EB3D 722F                <1>     jb      short m_pix_op_sub_2 ; 16bpp
8267                              <1>     ; 24 bit true colors
8268                              <1>     ;jmp    short m_pix_op_sub_24
8269                              <1> m_pix_op_sub_24:
8270                              <1>     ; 24 bit true colors
```

```
8271 0000EB3F 89C2                <1>     mov     edx, eax ; new color
8272                              <1>     ; 02/03/2021
8273                              <1> m_pix_op_sub_24_0:
8274 0000EB41 66AD                <1>     lodsw
8275 0000EB43 C1E010              <1>     shl     eax, 16
8276 0000EB46 AC                  <1>     lodsb
8277 0000EB47 C1C010              <1>     rol     eax, 16
8278 0000EB4A 3B05[6A9D0100]      <1>     cmp     eax, [maskcolor]
8279 0000EB50 7416                <1>     je      short m_pix_op_sub_24_2 ; exclude
8280 0000EB52 8305[348E0100]03    <1>     add     dword [u.r0], 3 ; +3
8281 0000EB59 29D0                <1>     sub     eax, edx
8282 0000EB5B 7302                <1>     jnb     short m_pix_op_sub_24_1
8283 0000EB5D 31C0                <1>     xor     eax, eax ; 0
8284                              <1> m_pix_op_sub_24_1:
8285 0000EB5F 668907              <1>     mov     [edi], ax
8286 0000EB62 C1E810              <1>     shr     eax, 16
8287 0000EB65 884702              <1>     mov     [edi+2], al
8288                              <1> m_pix_op_sub_24_2:
8289 0000EB68 83C703              <1>     add     edi, 3 ; +3
8290 0000EB6B E2D4                <1>     loop    m_pix_op_sub_24_0
8291 0000EB6D C3                  <1>     retn
8292                              <1>     ; 65536 colors (16bpp)
8293                              <1> m_pix_op_sub_2:
8294                              <1>     ;jmp    short m_pix_op_sub_16
8295                              <1> m_pix_op_sub_16:
8296                              <1>     ; 16 bit colors (65536 colors)
8297 0000EB6E 89C2                <1>     mov     edx, eax ; new color
8298                              <1> m_pix_op_sub_16_0:
8299 0000EB70 66AD                <1>     lodsw
8300 0000EB72 663B05[6A9D0100]    <1>     cmp     ax, [maskcolor]
8301 0000EB79 7411                <1>     je      short m_pix_op_sub_16_1 ; exclude
8302 0000EB7B 8305[348E0100]02    <1>     add     dword [u.r0], 2 ; +2
8303 0000EB82 662917              <1>     sub     [edi], dx
8304 0000EB85 7305                <1>     jnb     short m_pix_op_sub_16_1
8305 0000EB87 31C0                <1>     xor     eax, eax
8306 0000EB89 668907              <1>     mov     [edi], ax ; 0
8307                              <1> m_pix_op_sub_16_1:
8308 0000EB8C 83C702              <1>     add     edi, 2 ; +2
8309 0000EB8F E2DF                <1>     loop    m_pix_op_sub_16_0
8310 0000EB91 C3                  <1>     retn
8311                              <1> m_pix_op_sub_3:
8312                              <1>     ; 32 bit true colors
8313                              <1>     ;jmp    short m_pix_op_sub_32
8314                              <1> m_pix_op_sub_32:
8315                              <1>     ; 32 bit true colors
8316 0000EB92 89C2                <1>     mov     edx, eax ; new color
8317                              <1> m_pix_op_sub_32_0:
8318 0000EB94 AD                  <1>     lodsd
8319 0000EB95 3B05[6A9D0100]      <1>     cmp     eax, [maskcolor]
8320 0000EB9B 740F                <1>     je      short m_pix_op_sub_32_1 ; exclude
8321 0000EB9D 8305[348E0100]04    <1>     add     dword [u.r0], 4 ; +4
8322 0000EBA4 2917                <1>     sub     [edi], edx
8323 0000EBA6 7304                <1>     jnb     short m_pix_op_sub_32_1
8324 0000EBA8 31C0                <1>     xor     eax, eax
8325 0000EBAA 8907                <1>     mov     [edi], eax ; 0
8326                              <1> m_pix_op_sub_32_1:
8327 0000EBAC 83C704              <1>     add     edi, 4 ; +4
8328 0000EBAF E2E3                <1>     loop    m_pix_op_sub_32_0
8329 0000EBB1 C3                  <1>     retn
8330                              <1>
8331                              <1> m_pix_op_sub_w:
8332                              <1>     ; 06/02/2021
8333                              <1>     ; SUBTRACT COLOR (MASKED, window)
8334                              <1>     ;
8335                              <1>     ; jump from pix_op_sub_w
8336                              <1>     ;
8337                              <1>     ; INPUT:
8338                              <1>     ;   ecx = bytes per row (to be applied)
8339                              <1>     ;   edx = screen width in bytes
8340                              <1>     ;   ebx = row count
8341                              <1>     ;   eax = color
8342                              <1>     ;
8343                              <1>     ;   [maskcolor] = mask color (to be excluded)
8344                              <1>     ;
8345                              <1>     ; OUTPUT:
8346                              <1>     ;       [u.r0] will be > 0 if succesful
8347                              <1>
8348                              <1>     ; window
8349                              <1>     ;mov    edi, [v_str] ; LFB start address
8350                              <1>     ;mov    esi, edi
8351                              <1>
8352 0000EBB2 803D[599D0100]08    <1>     cmp     byte [v_bpp], 8 ; 8bpp
8353 0000EBB9 7707                <1>     ja      short m_pix_op_sub_w_1
8354                              <1>
8355                              <1>     ; 256 colors (8bpp)
8356 0000EBBB BD[18EB0000]        <1>     mov     ebp, m_pix_op_sub_8
8357 0000EBC0 EB1E                <1>     jmp     short m_pix_op_sub_w_4
8358                              <1>
8359                              <1> m_pix_op_sub_w_1:
8360 0000EBC2 803D[599D0100]18    <1>     cmp     byte [v_bpp], 24 ; 24bpp
8361 0000EBC9 7710                <1>     ja      short m_pix_op_sub_w_3 ; 32bpp
8362 0000EBCB 7207                <1>     jb      short m_pix_op_sub_w_2 ; 16bpp
8363                              <1>
8364                              <1>     ; 24 bit true colors
8365 0000EBCD BD[3FEB0000]        <1>     mov     ebp, m_pix_op_sub_24
8366 0000EBD2 EB0C                <1>     jmp     short m_pix_op_sub_w_4
8367                              <1>
8368                              <1>     ; 65536 colors (16bpp)
8369                              <1> m_pix_op_sub_w_2:
8370 0000EBD4 BD[6EEB0000]        <1>     mov     ebp, m_pix_op_sub_16
8371 0000EBD9 EB05                <1>     jmp     short m_pix_op_sub_w_4
8372                              <1>
8373                              <1>     ; 32 bit true colors
8374                              <1> m_pix_op_sub_w_3:
8375 0000EBDB BD[92EB0000]        <1>     mov     ebp, m_pix_op_sub_32
8376                              <1> m_pix_op_sub_w_4:
8377 0000EBE0 E927FEFFFF          <1>     jmp     m_pix_op_sub_w_x
8378                              <1>
8379                              <1> m_pix_op_mix:
8380                              <1>     ; 25/02/2021
8381                              <1>     ; 06/02/2021
8382                              <1>     ; MIX COLOR (MASKED, full screen)
8383                              <1>     ;
8384                              <1>     ; jump from pix_op_mix
8385                              <1>     ;
8386                              <1>     ; INPUT:
8387                              <1>     ;   eax = color (AL, AX, EAX)
8388                              <1>     ;   ecx = [v_siz] ; display page pixel count
8389                              <1>     ;   esi = edi = [v_mem] ; LFB start address
8390                              <1>     ;
8391                              <1>     ;   [maskcolor] = mask color (to be excluded)
8392                              <1>     ;
8393                              <1>     ; OUTPUT:
8394                              <1>     ;       [u.r0] will be > 0 if succesful
```

```
                                <1>
                                <1>     ; Full screen
                                <1> m_pix_op_mix_0:
0000EBE5 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000EBEC 771B                   <1>     ja      short m_pix_op_mix_1
                                <1>     ; 256 colors (8bpp)
                                <1>     ;jmp    short m_pix_op_mix_8
                                <1> m_pix_op_mix_8:
                                <1>     ; 8 bit colors (256 colors)
0000EBEE 88C2                   <1>     mov     dl, al ; new (mixing) color
                                <1> m_pix_op_mix_8_0:
0000EBF0 AC                     <1>     lodsb
0000EBF1 3A05[6A9D0100]         <1>     cmp     al, [maskcolor]
0000EBF7 740C                   <1>     je      short m_pix_op_mix_8_1 ; exclude
0000EBF9 00D0                   <1>     add     al, dl ; 25/02/2021
0000EBFB D0D8                   <1>     rcr     al, 1
0000EBFD 8807                   <1>     mov     [edi], al
0000EBFF FF05[348E0100]         <1>     inc     dword [u.r0] ; +1
                                <1> m_pix_op_mix_8_1:
0000EC05 47                     <1>     inc     edi
0000EC06 E2E8                   <1>     loop    m_pix_op_mix_8_0
0000EC08 C3                     <1>     retn
                                <1> m_pix_op_mix_1:
0000EC09 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000EC10 7752                   <1>     ja      short m_pix_op_mix_3 ; 32bpp
0000EC12 722D                   <1>     jb      short m_pix_op_mix_2 ; 16bpp
                                <1>     ; 24 bit true colors
                                <1>     ;jmp    short m_pix_op_mix_24
                                <1> m_pix_op_mix_24:
                                <1>     ; 24 bit true colors
0000EC14 89C2                   <1>     mov     edx, eax ; new color
                                <1>     ;and    edx, 0FFFFFFh
                                <1> m_pix_op_mix_24_0:
0000EC16 66AD                   <1>     lodsw
0000EC18 C1E010                 <1>     shl     eax, 16
0000EC1B AC                     <1>     lodsb
0000EC1C C1C010                 <1>     rol     eax, 16
0000EC1F 3B05[6A9D0100]         <1>     cmp     eax, [maskcolor]
0000EC25 7414                   <1>     je      short m_pix_op_mix_24_1 ; exclude
0000EC27 01D0                   <1>     add     eax, edx
0000EC29 D1E8                   <1>     shr     eax, 1
0000EC2B 668907                 <1>     mov     [edi], ax
0000EC2E C1E810                 <1>     shr     eax, 16
0000EC31 884702                 <1>     mov     [edi+2], al
0000EC34 8305[348E0100]03       <1>     add     dword [u.r0], 3 ; +3
                                <1> m_pix_op_mix_24_1:
0000EC3B 83C703                 <1>     add     edi, 3 ; +3
0000EC3E E2D6                   <1>     loop    m_pix_op_mix_24_0
0000EC40 C3                     <1>     retn
                                <1>     ; 65536 colors (16bpp)
                                <1> m_pix_op_mix_2:
                                <1>     ;jmp    short m_pix_op_mix_16
                                <1> m_pix_op_mix_16:
                                <1>     ; 16 bit colors (65536 colors)
0000EC41 89C2                   <1>     mov     edx, eax ; new color
                                <1>     ;and    edx, 0FFFFh
                                <1> m_pix_op_mix_16_0:
0000EC43 66AD                   <1>     lodsw
0000EC45 663B05[6A9D0100]       <1>     cmp     ax, [maskcolor]
0000EC4C 7410                   <1>     je      short m_pix_op_mix_16_1 ; exclude
0000EC4E 6601D0                 <1>     add     ax, dx
0000EC51 66D1D8                 <1>     rcr     ax, 1
0000EC54 668907                 <1>     mov     [edi], ax
0000EC57 8305[348E0100]02       <1>     add     dword [u.r0], 2 ; +2
                                <1> m_pix_op_mix_16_1:
0000EC5E 83C702                 <1>     add     edi, 2 ; +2
0000EC61 E2E0                   <1>     loop    m_pix_op_mix_16_0
0000EC63 C3                     <1>     retn
                                <1> m_pix_op_mix_3:
                                <1>     ; 32 bit true colors
                                <1>     ;jmp    short m_pix_op_mix_32
                                <1> m_pix_op_mix_32:
                                <1>     ; 32 bit true colors
0000EC64 89C2                   <1>     mov     edx, eax ; new color
                                <1> m_pix_op_mix_32_0:
0000EC66 AD                     <1>     lodsd
0000EC67 3B05[6A9D0100]         <1>     cmp     eax, [maskcolor]
0000EC6D 740D                   <1>     je      short m_pix_op_mix_32_1 ; exclude
0000EC6F 01D0                   <1>     add     eax, edx
                                <1>     ; 02/03/2021
0000EC71 D1D8                   <1>     rcr     eax, 1
0000EC73 8907                   <1>     mov     [edi], eax
0000EC75 8305[348E0100]04       <1>     add     dword [u.r0], 4 ; +4
                                <1> m_pix_op_mix_32_1:
0000EC7C 83C704                 <1>     add     edi, 4 ; +4
0000EC7F E2E5                   <1>     loop    m_pix_op_mix_32_0
0000EC81 C3                     <1>     retn
                                <1>
                                <1> m_pix_op_mix_w:
                                <1>     ; 06/02/2021
                                <1>     ; MIX COLOR (MASKED, window)
                                <1>     ;
                                <1>     ; jump from pix_op_mix_w
                                <1>     ;
                                <1>     ; INPUT:
                                <1>     ;   ecx = bytes per row (to be applied)
                                <1>     ;   edx = screen width in bytes
                                <1>     ;   ebx = row count
                                <1>     ;   eax = color
                                <1>     ;
                                <1>     ;   [maskcolor] = mask color (to be excluded)
                                <1>     ;
                                <1>     ; OUTPUT:
                                <1>     ;       [u.r0] will be > 0 if succesful
                                <1>
                                <1>     ; window
                                <1>     ;mov    edi, [v_str] ; LFB start address
                                <1>     ;mov    esi, edi
                                <1>
0000EC82 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000EC89 7707                   <1>     ja      short m_pix_op_mix_w_1
                                <1>
                                <1>     ; 256 colors (8bpp)
0000EC8B BD[EEEB0000]           <1>     mov     ebp, m_pix_op_mix_8
0000EC90 EB1E                   <1>     jmp     short m_pix_op_mix_w_4
                                <1>
                                <1> m_pix_op_mix_w_1:
0000EC92 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000EC99 7710                   <1>     ja      short m_pix_op_mix_w_3 ; 32bpp
0000EC9B 7207                   <1>     jb      short m_pix_op_mix_w_2 ; 16bpp
                                <1>
                                <1>     ; 24 bit true colors
0000EC9D BD[14EC0000]           <1>     mov     ebp, m_pix_op_mix_24
0000ECA2 EB0C                   <1>     jmp     short m_pix_op_mix_w_4
```

```
                                <1>
                                <1>     ; 65536 colors (16bpp)
                                <1> m_pix_op_mix_w_2:
0000ECA4 BD[41EC0000]           <1>     mov     ebp, m_pix_op_mix_16
0000ECA9 EB05                   <1>     jmp     short m_pix_op_mix_w_4
                                <1>
                                <1>     ; 32 bit true colors
                                <1> m_pix_op_mix_w_3:
0000ECAB BD[64EC0000]           <1>     mov     ebp, m_pix_op_mix_32
                                <1> m_pix_op_mix_w_4:
0000ECB0 E957FDFFFF             <1>     jmp     m_pix_op_mix_w_x
                                <1>
                                <1> m_pix_op_and:
                                <1>     ; 06/02/2021
                                <1>     ; AND COLOR (MASKED, full screen)
                                <1>     ;
                                <1>     ; jump from pix_op_and
                                <1>     ;
                                <1>     ; INPUT:
                                <1>     ;   eax = color (AL, AX, EAX)
                                <1>     ;   ecx = [v_siz] ; display page pixel count
                                <1>     ;   esi = edi = [v_mem] ; LFB start address
                                <1>     ;
                                <1>     ;   [maskcolor] = mask color (to be excluded)
                                <1>     ;
                                <1>     ; OUTPUT:
                                <1>     ;       [u.r0] will be > 0 if succesful
                                <1>
                                <1>     ; Full screen
                                <1> m_pix_op_and_0:
0000ECB5 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000ECBC 7717                   <1>     ja      short m_pix_op_and_1
                                <1>     ; 256 colors (8bpp)
                                <1>     ;jmp    short m_pix_op_and_8
                                <1> m_pix_op_and_8:
                                <1>     ; 8 bit colors (256 colors)
0000ECBE 88C2                   <1>     mov     dl, al ; new color
                                <1> m_pix_op_and_8_0:
0000ECC0 AC                     <1>     lodsb
0000ECC1 3A05[6A9D0100]         <1>     cmp     al, [maskcolor]
0000ECC7 7408                   <1>     je      short m_pix_op_and_8_1 ; exclude
0000ECC9 2017                   <1>     and     [edi], dl
0000ECCB FF05[348E0100]         <1>     inc     dword [u.r0] ; +1
                                <1> m_pix_op_and_8_1:
0000ECD1 47                     <1>     inc     edi
0000ECD2 E2EC                   <1>     loop    m_pix_op_and_8_0
0000ECD4 C3                     <1>     retn
                                <1> m_pix_op_and_1:
0000ECD5 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000ECDC 774A                   <1>     ja      short m_pix_op_and_3 ; 32bpp
0000ECDE 722B                   <1>     jb      short m_pix_op_and_2 ; 16bpp
                                <1>     ; 24 bit true colors
                                <1>     ;jmp    short m_pix_op_and_24
                                <1> m_pix_op_and_24:
                                <1>     ; 24 bit true colors
0000ECE0 89C2                   <1>     mov     edx, eax ; new color
                                <1>     ;and    edx, 0FFFFFFh
                                <1> m_pix_op_and_24_0:
0000ECE2 66AD                   <1>     lodsw
0000ECE4 C1E010                 <1>     shl     eax, 16
0000ECE7 AC                     <1>     lodsb
0000ECE8 C1C010                 <1>     rol     eax, 16
0000ECEB 3B05[6A9D0100]         <1>     cmp     eax, [maskcolor]
0000ECF1 7412                   <1>     je      short m_pix_op_and_24_1 ; exclude
0000ECF3 21D0                   <1>     and     eax, edx
0000ECF5 668907                 <1>     mov     [edi], ax
0000ECF8 C1E810                 <1>     shr     eax, 16
0000ECFB 884702                 <1>     mov     [edi+2], al
0000ECFE 8305[348E0100]03       <1>     add     dword [u.r0], 3 ; +3
                                <1> m_pix_op_and_24_1:
0000ED05 83C703                 <1>     add     edi, 3 ; +3
0000ED08 E2D8                   <1>     loop    m_pix_op_and_24_0
0000ED0A C3                     <1>     retn
                                <1>     ; 65536 colors (16bpp)
                                <1> m_pix_op_and_2:
                                <1>     ;jmp    short m_pix_op_and_16
                                <1> m_pix_op_and_16:
                                <1>     ; 16 bit colors (65536 colors)
0000ED0B 89C2                   <1>     mov     edx, eax ; new color
                                <1>     ;and    edx, 0FFFFh
                                <1> m_pix_op_and_16_0:
0000ED0D 66AD                   <1>     lodsw
0000ED0F 663B05[6A9D0100]       <1>     cmp     ax, [maskcolor]
0000ED16 740A                   <1>     je      short m_pix_op_and_16_1 ; exclude
0000ED18 662117                 <1>     and     [edi], dx
0000ED1B 8305[348E0100]02       <1>     add     dword [u.r0], 2 ; +2
                                <1> m_pix_op_and_16_1:
0000ED22 83C702                 <1>     add     edi, 2 ; +2
0000ED25 E2E6                   <1>     loop    m_pix_op_and_16_0
0000ED27 C3                     <1>     retn
                                <1> m_pix_op_and_3:
                                <1>     ; 32 bit true colors
                                <1>     ;jmp    short m_pix_op_and_32
                                <1> m_pix_op_and_32:
                                <1>     ; 32 bit true colors
0000ED28 89C2                   <1>     mov     edx, eax ; new color
                                <1> m_pix_op_and_32_0:
0000ED2A AD                     <1>     lodsd
0000ED2B 3B05[6A9D0100]         <1>     cmp     eax, [maskcolor]
0000ED31 7409                   <1>     je      short m_pix_op_and_32_1 ; exclude
0000ED33 2117                   <1>     and     [edi], edx ; 25/02/2021
0000ED35 8305[348E0100]04       <1>     add     dword [u.r0], 4 ; +4
                                <1> m_pix_op_and_32_1:
0000ED3C 83C704                 <1>     add     edi, 4 ; +4
0000ED3F E2E9                   <1>     loop    m_pix_op_and_32_0
0000ED41 C3                     <1>     retn
                                <1>
                                <1> m_pix_op_and_w:
                                <1>     ; 06/02/2021
                                <1>     ; AND COLOR (MASKED, window)
                                <1>     ;
                                <1>     ; jump from pix_op_and_w
                                <1>     ;
                                <1>     ; INPUT:
                                <1>     ;   ecx = bytes per row (to be applied)
                                <1>     ;   edx = screen width in bytes
                                <1>     ;   ebx = row count
                                <1>     ;   eax = color
                                <1>     ;
                                <1>     ;   [maskcolor] = mask color (to be excluded)
                                <1>     ;
                                <1>     ; OUTPUT:
                                <1>     ;       [u.r0] will be > 0 if succesful
                                <1>
```

```
                                <1>     ; window
                                <1>     ;mov    edi, [v_str] ; LFB start address
                                <1>     ;mov    esi, edi
                                <1>
0000ED42 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000ED49 7707                   <1>     ja      short m_pix_op_and_w_1
                                <1>
                                <1>     ; 256 colors (8bpp)
0000ED4B BD[BEEC0000]           <1>     mov     ebp, m_pix_op_and_8
0000ED50 EB1E                   <1>     jmp     short m_pix_op_and_w_4
                                <1>
                                <1> m_pix_op_and_w_1:
0000ED52 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000ED59 7710                   <1>     ja      short m_pix_op_and_w_3 ; 32bpp
0000ED5B 7207                   <1>     jb      short m_pix_op_and_w_2 ; 16bpp
                                <1>
                                <1>     ; 24 bit true colors
0000ED5D BD[E0EC0000]           <1>     mov     ebp, m_pix_op_and_24
0000ED62 EB0C                   <1>     jmp     short m_pix_op_and_w_4
                                <1>
                                <1>     ; 65536 colors (16bpp)
                                <1> m_pix_op_and_w_2:
0000ED64 BD[0BED0000]           <1>     mov     ebp, m_pix_op_and_16
0000ED69 EB05                   <1>     jmp     short m_pix_op_and_w_4
                                <1>
                                <1>     ; 32 bit true colors
                                <1> m_pix_op_and_w_3:
0000ED6B BD[28ED0000]           <1>     mov     ebp, m_pix_op_and_32
                                <1> m_pix_op_and_w_4:
0000ED70 E997FCFFFF             <1>     jmp     m_pix_op_and_w_x
                                <1>
                                <1> m_pix_op_or:
                                <1>     ; 06/02/2021
                                <1>     ; OR COLOR (MASKED, full screen)
                                <1>     ;
                                <1>     ; jump from pix_op_orc
                                <1>     ;
                                <1>     ; INPUT:
                                <1>     ;   eax = color (AL, AX, EAX)
                                <1>     ;   ecx = [v_siz] ; display page pixel count
                                <1>     ;   esi = edi = [v_mem] ; LFB start address
                                <1>     ;
                                <1>     ;   [maskcolor] = mask color (to be excluded)
                                <1>     ;
                                <1>     ; OUTPUT:
                                <1>     ;       [u.r0] will be > 0 if succesful
                                <1>
                                <1>     ; Full screen
                                <1> m_pix_op_or_0:
0000ED75 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000ED7C 7717                   <1>     ja      short m_pix_op_or_1
                                <1>     ; 256 colors (8bpp)
                                <1>     ;jmp    short m_pix_op_or_8
                                <1> m_pix_op_or_8:
                                <1>     ; 8 bit colors (256 colors)
0000ED7E 88C2                   <1>     mov     dl, al ; new color
                                <1> m_pix_op_or_8_0:
0000ED80 AC                     <1>     lodsb
0000ED81 3A05[6A9D0100]         <1>     cmp     al, [maskcolor]
0000ED87 7408                   <1>     je      short m_pix_op_or_8_1 ; exclude
0000ED89 0817                   <1>     or      [edi], dl
0000ED8B FF05[348E0100]         <1>     inc     dword [u.r0] ; +1
                                <1> m_pix_op_or_8_1:
0000ED91 47                     <1>     inc     edi
0000ED92 E2EC                   <1>     loop    m_pix_op_or_8_0
0000ED94 C3                     <1>     retn
                                <1> m_pix_op_or_1:
0000ED95 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000ED9C 774A                   <1>     ja      short m_pix_op_or_3 ; 32bpp
0000ED9E 722B                   <1>     jb      short m_pix_op_or_2 ; 16bpp
                                <1>     ; 24 bit true colors
                                <1>     ;jmp    short m_pix_op_or_24
                                <1> m_pix_op_or_24:
                                <1>     ; 24 bit true colors
0000EDA0 89C2                   <1>     mov     edx, eax ; new color
                                <1>     ;and    edx, 0FFFFFFh
                                <1> m_pix_op_or_24_0:
0000EDA2 66AD                   <1>     lodsw
0000EDA4 C1E010                 <1>     shl     eax, 16
0000EDA7 AC                     <1>     lodsb
0000EDA8 C1C010                 <1>     rol     eax, 16
0000EDAB 3B05[6A9D0100]         <1>     cmp     eax, [maskcolor]
0000EDB1 7412                   <1>     je      short m_pix_op_or_24_1 ; exclude
0000EDB3 09D0                   <1>     or      eax, edx
0000EDB5 668907                 <1>     mov     [edi], ax
0000EDB8 C1E810                 <1>     shr     eax, 16
0000EDBB 884702                 <1>     mov     [edi+2], al
0000EDBE 8305[348E0100]03       <1>     add     dword [u.r0], 3 ; +3
                                <1> m_pix_op_or_24_1:
0000EDC5 83C703                 <1>     add     edi, 3 ; +3
0000EDC8 E2D8                   <1>     loop    m_pix_op_or_24_0
0000EDCA C3                     <1>     retn
                                <1>     ; 65536 colors (16bpp)
                                <1> m_pix_op_or_2:
                                <1>     ;jmp    short m_pix_op_or_16
                                <1> m_pix_op_or_16:
                                <1>     ; 16 bit colors (65536 colors)
0000EDCB 89C2                   <1>     mov     edx, eax ; new color
                                <1>     ;and    edx, 0FFFFh
                                <1> m_pix_op_or_16_0:
0000EDCD 66AD                   <1>     lodsw
0000EDCF 663B05[6A9D0100]       <1>     cmp     ax, [maskcolor]
0000EDD6 740A                   <1>     je      short m_pix_op_or_16_1 ; exclude
0000EDD8 660917                 <1>     or      [edi], dx
0000EDDB 8305[348E0100]02       <1>     add     dword [u.r0], 2 ; +2
                                <1> m_pix_op_or_16_1:
0000EDE2 83C702                 <1>     add     edi, 2 ; +2
0000EDE5 E2E6                   <1>     loop    m_pix_op_or_16_0
0000EDE7 C3                     <1>     retn
                                <1> m_pix_op_or_3:
                                <1>     ; 32 bit true colors
                                <1>     ;jmp    short m_pix_op_or_32
                                <1> m_pix_op_or_32:
                                <1>     ; 32 bit true colors
0000EDE8 89C2                   <1>     mov     edx, eax ; new color
                                <1> m_pix_op_or_32_0:
0000EDEA AD                     <1>     lodsd
0000EDEB 3B05[6A9D0100]         <1>     cmp     eax, [maskcolor]
0000EDF1 7409                   <1>     je      short m_pix_op_or_32_1 ; exclude
0000EDF3 0917                   <1>     or      [edi], edx ; 25/02/2021
0000EDF5 8305[348E0100]04       <1>     add     dword [u.r0], 4 ; +4
                                <1> m_pix_op_or_32_1:
0000EDFC 83C704                 <1>     add     edi, 4 ; +4
0000EDFF E2E9                   <1>     loop    m_pix_op_or_32_0
```

```
8767 0000EE01 C3                <1>     retn
8768                            <1>
8769                            <1> m_pix_op_or_w:
8770                            <1>     ; 06/02/2021
8771                            <1>     ; MIX COLOR (MASKED, window)
8772                            <1>     ;
8773                            <1>     ; jump from pix_op_or_w
8774                            <1>     ;
8775                            <1>     ; INPUT:
8776                            <1>     ;   ecx = bytes per row (to be applied)
8777                            <1>     ;   edx = screen width in bytes
8778                            <1>     ;   ebx = row count
8779                            <1>     ;   eax = color
8780                            <1>     ;
8781                            <1>     ;   [maskcolor] = mask color (to be excluded)
8782                            <1>     ;
8783                            <1>     ; OUTPUT:
8784                            <1>     ;       [u.r0] will be > 0 if succesful
8785                            <1>
8786                            <1>     ; window
8787                            <1>     ;mov    edi, [v_str] ; LFB start address
8788                            <1>     ;mov    esi, edi
8789                            <1>
8790 0000EE02 803D[599D0100]08 <1>     cmp     byte [v_bpp], 8 ; 8bpp
8791 0000EE09 7707             <1>     ja      short m_pix_op_or_w_1
8792                            <1>
8793                            <1>     ; 256 colors (8bpp)
8794 0000EE0B BD[7EED0000]     <1>     mov     ebp, m_pix_op_or_8
8795 0000EE10 EB1E             <1>     jmp     short m_pix_op_or_w_4
8796                            <1>
8797                            <1> m_pix_op_or_w_1:
8798 0000EE12 803D[599D0100]18 <1>     cmp     byte [v_bpp], 24 ; 24bpp
8799 0000EE19 7710             <1>     ja      short m_pix_op_or_w_3 ; 32bpp
8800 0000EE1B 7207             <1>     jb      short m_pix_op_or_w_2 ; 16bpp
8801                            <1>
8802                            <1>     ; 24 bit true colors
8803 0000EE1D BD[A0ED0000]     <1>     mov     ebp, m_pix_op_or_24
8804 0000EE22 EB0C             <1>     jmp     short m_pix_op_or_w_4
8805                            <1>
8806                            <1>     ; 65536 colors (16bpp)
8807                            <1> m_pix_op_or_w_2:
8808 0000EE24 BD[CBED0000]     <1>     mov     ebp, m_pix_op_or_16
8809 0000EE29 EB05             <1>     jmp     short m_pix_op_or_w_4
8810                            <1>
8811                            <1>     ; 32 bit true colors
8812                            <1> m_pix_op_or_w_3:
8813 0000EE2B BD[E8ED0000]     <1>     mov     ebp, m_pix_op_or_32
8814                            <1> m_pix_op_or_w_4:
8815 0000EE30 E9D7FBFFFF       <1>     jmp     m_pix_op_orc_w_x
8816                            <1>
8817                            <1> m_pix_op_xor:
8818                            <1>     ; 06/02/2021
8819                            <1>     ; XOR COLOR (MASKED, full screen)
8820                            <1>     ;
8821                            <1>     ; jump from pix_op_xor
8822                            <1>     ;
8823                            <1>     ; INPUT:
8824                            <1>     ;   eax = color (AL, AX, EAX)
8825                            <1>     ;   ecx = [v_siz] ; display page pixel count
8826                            <1>     ;   esi = edi = [v_mem] ; LFB start address
8827                            <1>     ;
8828                            <1>     ;   [maskcolor] = mask color (to be excluded)
8829                            <1>     ;
8830                            <1>     ; OUTPUT:
8831                            <1>     ;       [u.r0] will be > 0 if succesful
8832                            <1>
8833                            <1>     ; Full screen
8834                            <1> m_pix_op_xor_0:
8835 0000EE35 803D[599D0100]08 <1>     cmp     byte [v_bpp], 8 ; 8bpp
8836 0000EE3C 7717             <1>     ja      short m_pix_op_xor_1
8837                            <1>     ; 256 colors (8bpp)
8838                            <1>     ;jmp    short m_pix_op_xor_8
8839                            <1> m_pix_op_xor_8:
8840                            <1>     ; 8 bit colors (256 colors)
8841 0000EE3E 88C2              <1>     mov     dl, al ; new color
8842                            <1> m_pix_op_xor_8_0:
8843 0000EE40 AC               <1>     lodsb
8844 0000EE41 3A05[6A9D0100]   <1>     cmp     al, [maskcolor]
8845 0000EE47 7408             <1>     je      short m_pix_op_xor_8_1 ; exclude
8846 0000EE49 3017             <1>     xor     [edi], dl
8847 0000EE4B FF05[348E0100]   <1>     inc     dword [u.r0] ; +1
8848                            <1> m_pix_op_xor_8_1:
8849 0000EE51 47               <1>     inc     edi
8850 0000EE52 E2EC             <1>     loop    m_pix_op_xor_8_0
8851 0000EE54 C3               <1>     retn
8852                            <1> m_pix_op_xor_1:
8853 0000EE55 803D[599D0100]18 <1>     cmp     byte [v_bpp], 24 ; 24bpp
8854 0000EE5C 774A             <1>     ja      short m_pix_op_xor_3 ; 32bpp
8855 0000EE5E 722B             <1>     jb      short m_pix_op_xor_2 ; 16bpp
8856                            <1>     ; 24 bit true colors
8857                            <1>     ;jmp    short m_pix_op_xor_24
8858                            <1> m_pix_op_xor_24:
8859                            <1>     ; 24 bit true colors
8860 0000EE60 89C2             <1>     mov     edx, eax ; new color
8861                            <1>     ;and    edx, 0FFFFFFh
8862                            <1> m_pix_op_xor_24_0:
8863 0000EE62 66AD             <1>     lodsw
8864 0000EE64 C1E010           <1>     shl     eax, 16
8865 0000EE67 AC               <1>     lodsb
8866 0000EE68 C1C010           <1>     rol     eax, 16
8867 0000EE6B 3B05[6A9D0100]   <1>     cmp     eax, [maskcolor]
8868 0000EE71 7412             <1>     je      short m_pix_op_xor_24_1 ; exclude
8869 0000EE73 31D0             <1>     xor     eax, edx
8870 0000EE75 668907           <1>     mov     [edi], ax
8871 0000EE78 C1E810           <1>     shr     eax, 16
8872 0000EE7B 884702           <1>     mov     [edi+2], al
8873 0000EE7E 8305[348E0100]03 <1>     add     dword [u.r0], 3 ; +3
8874                            <1> m_pix_op_xor_24_1:
8875 0000EE85 83C703           <1>     add     edi, 3 ; +3
8876 0000EE88 E2D8             <1>     loop    m_pix_op_xor_24_0
8877 0000EE8A C3               <1>     retn
8878                            <1>     ; 65536 colors (16bpp)
8879                            <1> m_pix_op_xor_2:
8880                            <1>     ;jmp    short m_pix_op_xor_16
8881                            <1> m_pix_op_xor_16:
8882                            <1>     ; 16 bit colors (65536 colors)
8883 0000EE8B 89C2             <1>     mov     edx, eax ; new color
8884                            <1>     ;and    edx, 0FFFFh
8885                            <1> m_pix_op_xor_16_0:
8886 0000EE8D 66AD             <1>     lodsw
8887 0000EE8F 663B05[6A9D0100] <1>     cmp     ax, [maskcolor]
8888 0000EE96 740A             <1>     je      short m_pix_op_xor_16_1 ; exclude
8889 0000EE98 663117           <1>     xor     [edi], dx
8890 0000EE9B 8305[348E0100]02 <1>     add     dword [u.r0], 2 ; +2
```

```
                                <1> m_pix_op_xor_16_1:
0000EEA2 83C702                 <1>     add     edi, 2 ; +2
0000EEA5 E2E6                   <1>     loop    m_pix_op_xor_16_0
0000EEA7 C3                     <1>     retn
                                <1> m_pix_op_xor_3:
                                <1>     ; 32 bit true colors
                                <1>     ;jmp    short m_pix_op_xor_32
                                <1> m_pix_op_xor_32:
                                <1>     ; 32 bit true colors
0000EEA8 89C2                   <1>     mov     edx, eax ; new color
                                <1> m_pix_op_xor_32_0:
0000EEAA AD                     <1>     lodsd
0000EEAB 3B05[6A9D0100]         <1>     cmp     eax, [maskcolor]
0000EEB1 7409                   <1>     je      short m_pix_op_xor_32_1 ; exclude
0000EEB3 3117                   <1>     xor     [edi], edx ; 25/02/2021
0000EEB5 8305[348E0100]04       <1>     add     dword [u.r0], 4 ; +4
                                <1> m_pix_op_xor_32_1:
0000EEBC 83C704                 <1>     add     edi, 4 ; +4
0000EEBF E2E9                   <1>     loop    m_pix_op_xor_32_0
0000EEC1 C3                     <1>     retn
                                <1>
                                <1> m_pix_op_xor_w:
                                <1>     ; 06/02/2021
                                <1>     ; XOR COLOR (MASKED, window)
                                <1>     ;
                                <1>     ; jump from pix_op_xor_w
                                <1>     ;
                                <1>     ; INPUT:
                                <1>     ;   ecx = bytes per row (to be applied)
                                <1>     ;   edx = screen width in bytes
                                <1>     ;   ebx = row count
                                <1>     ;   eax = color
                                <1>     ;
                                <1>     ;   [maskcolor] = mask color (to be excluded)
                                <1>     ;
                                <1>     ; OUTPUT:
                                <1>     ;       [u.r0] will be > 0 if succesful
                                <1>
                                <1>     ; window
                                <1>     ;mov    edi, [v_str] ; LFB start address
                                <1>     ;mov    esi, edi
                                <1>
0000EEC2 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000EEC9 7707                   <1>     ja      short m_pix_op_xor_w_1
                                <1>
                                <1>     ; 256 colors (8bpp)
0000EECB BD[3EEE0000]           <1>     mov     ebp, m_pix_op_xor_8
0000EED0 EB1E                   <1>     jmp     short m_pix_op_xor_w_4
                                <1>
                                <1> m_pix_op_xor_w_1:
0000EED2 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000EED9 7710                   <1>     ja      short m_pix_op_xor_w_3 ; 32bpp
0000EEDB 7207                   <1>     jb      short m_pix_op_xor_w_2 ; 16bpp
                                <1>
                                <1>     ; 24 bit true colors
0000EEDD BD[60EE0000]           <1>     mov     ebp, m_pix_op_xor_24
0000EEE2 EB0C                   <1>     jmp     short m_pix_op_xor_w_4
                                <1>
                                <1>     ; 65536 colors (16bpp)
                                <1> m_pix_op_xor_w_2:
0000EEE4 BD[8BEE0000]           <1>     mov     ebp, m_pix_op_xor_16
0000EEE9 EB05                   <1>     jmp     short m_pix_op_xor_w_4
                                <1>
                                <1>     ; 32 bit true colors
                                <1> m_pix_op_xor_w_3:
0000EEEB BD[A8EE0000]           <1>     mov     ebp, m_pix_op_xor_32
                                <1> m_pix_op_xor_w_4:
0000EEF0 E917FBFFFF             <1>     jmp     m_pix_op_xor_w_x
                                <1>
                                <1> m_pix_op_not:
                                <1>     ; 06/02/2021
                                <1>     ; NOT COLOR (MASKED, full screen)
                                <1>     ;
                                <1>     ; jump from pix_op_not
                                <1>     ;
                                <1>     ; INPUT:
                                <1>     ;   ecx = [v_siz] ; display page pixel count
                                <1>     ;   esi = edi = [v_mem] ; LFB start address
                                <1>     ;
                                <1>     ;   [maskcolor] = mask color (to be excluded)
                                <1>     ;
                                <1>     ; OUTPUT:
                                <1>     ;       [u.r0] will be > 0 if succesful
                                <1>
                                <1>     ; Full screen
                                <1> m_pix_op_not_0:
0000EEF5 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000EEFC 7715                   <1>     ja      short m_pix_op_not_1
                                <1>     ; 256 colors (8bpp)
                                <1>     ;jmp    short m_pix_op_not_8
                                <1> m_pix_op_not_8:
                                <1>     ; 8 bit colors (256 colors)
0000EEFE AC                     <1>     lodsb
0000EEFF 3A05[6A9D0100]         <1>     cmp     al, [maskcolor]
0000EF05 7408                   <1>     je      short m_pix_op_not_8_1 ; exclude
0000EF07 F617                   <1>     not     byte [edi]
0000EF09 FF05[348E0100]         <1>     inc     dword [u.r0] ; +1
                                <1> m_pix_op_not_8_1:
0000EF0F 47                     <1>     inc     edi
0000EF10 E2EC                   <1>     loop    m_pix_op_not_8
0000EF12 C3                     <1>     retn
                                <1> m_pix_op_not_1:
0000EF13 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000EF1A 7746                   <1>     ja      short m_pix_op_not_3 ; 32bpp
0000EF1C 7229                   <1>     jb      short m_pix_op_not_2 ; 16bpp
                                <1>     ; 24 bit true colors
                                <1>     ;jmp    short m_pix_op_not_24
                                <1> m_pix_op_not_24:
                                <1>     ; 24 bit true colors
0000EF1E 66AD                   <1>     lodsw
0000EF20 C1E010                 <1>     shl     eax, 16
0000EF23 AC                     <1>     lodsb
0000EF24 C1C010                 <1>     rol     eax, 16
0000EF27 3B05[6A9D0100]         <1>     cmp     eax, [maskcolor]
0000EF2D 7412                   <1>     je      short m_pix_op_not_24_1 ; exclude
0000EF2F F7D0                   <1>     not     eax
0000EF31 668907                 <1>     mov     [edi], ax
0000EF34 C1E810                 <1>     shr     eax, 16
0000EF37 884702                 <1>     mov     [edi+2], al
0000EF3A 8305[348E0100]03       <1>     add     dword [u.r0], 3 ; +3
                                <1> m_pix_op_not_24_1:
0000EF41 83C703                 <1>     add     edi, 3 ; +3
0000EF44 E2D8                   <1>     loop    m_pix_op_not_24
0000EF46 C3                     <1>     retn
```

```
                                <1>     ; 65536 colors (16bpp)
                                <1> m_pix_op_not_2:
                                <1>     ;jmp    short m_pix_op_not_16
                                <1> m_pix_op_not_16:
                                <1>     ; 16 bit colors (65536 colors)
0000EF47 66AD                   <1>     lodsw
0000EF49 663B05[6A9D0100]       <1>     cmp     ax, [maskcolor]
0000EF50 740A                   <1>     je      short m_pix_op_not_16_1 ; exclude
0000EF52 66F717                 <1>     not     word [edi]
0000EF55 8305[348E0100]02       <1>     add     dword [u.r0], 2 ; +2
                                <1> m_pix_op_not_16_1:
0000EF5C 83C702                 <1>     add     edi, 2 ; +2
0000EF5F E2E6                   <1>     loop    m_pix_op_not_16
0000EF61 C3                     <1>     retn
                                <1> m_pix_op_not_3:
                                <1>     ; 32 bit true colors
                                <1>     ;jmp    short m_pix_op_not_32
                                <1> m_pix_op_not_32:
                                <1>     ; 32 bit true colors
0000EF62 AD                     <1>     lodsd
0000EF63 3B05[6A9D0100]         <1>     cmp     eax, [maskcolor]
0000EF69 7409                   <1>     je      short m_pix_op_not_32_1 ; exclude
0000EF6B F717                   <1>     not     dword [edi]
0000EF6D 8305[348E0100]04       <1>     add     dword [u.r0], 4 ; +4
                                <1> m_pix_op_not_32_1:
0000EF74 83C704                 <1>     add     edi, 4 ; +4
0000EF77 E2E9                   <1>     loop    m_pix_op_not_32
0000EF79 C3                     <1>     retn
                                <1>
                                <1> m_pix_op_not_w:
                                <1>     ; 06/02/2021
                                <1>     ; NOT COLOR (MASKED, window)
                                <1>     ;
                                <1>     ; jump from pix_op_not_w
                                <1>     ;
                                <1>     ; INPUT:
                                <1>     ;   ecx = bytes per row (to be applied)
                                <1>     ;   edx = screen width in bytes
                                <1>     ;   ebx = row count
                                <1>     ;
                                <1>     ;   [maskcolor] = mask color (to be excluded)
                                <1>     ;
                                <1>     ; OUTPUT:
                                <1>     ;       [u.r0] will be > 0 if succesful
                                <1>
                                <1>     ; window
                                <1>     ;mov    edi, [v_str] ; LFB start address
                                <1>     ;mov    esi, edi
                                <1>
0000EF7A 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000EF81 7707                   <1>     ja      short m_pix_op_not_w_1
                                <1>
                                <1>     ; 256 colors (8bpp)
0000EF83 BD[FEEE0000]           <1>     mov     ebp, m_pix_op_not_8
0000EF88 EB1E                   <1>     jmp     short m_pix_op_not_w_4
                                <1>
                                <1> m_pix_op_not_w_1:
0000EF8A 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000EF91 7710                   <1>     ja      short m_pix_op_not_w_3 ; 32bpp
0000EF93 7207                   <1>     jb      short m_pix_op_not_w_2 ; 16bpp
                                <1>
                                <1>     ; 24 bit true colors
0000EF95 BD[1EEF0000]           <1>     mov     ebp, m_pix_op_not_24
0000EF9A EB0C                   <1>     jmp     short m_pix_op_not_w_4
                                <1>
                                <1>     ; 65536 colors (16bpp)
                                <1> m_pix_op_not_w_2:
0000EF9C BD[47EF0000]           <1>     mov     ebp, m_pix_op_not_16
0000EFA1 EB05                   <1>     jmp     short m_pix_op_not_w_4
                                <1>
                                <1>     ; 32 bit true colors
                                <1> m_pix_op_not_w_3:
0000EFA3 BD[62EF0000]           <1>     mov     ebp, m_pix_op_not_32
                                <1> m_pix_op_not_w_4:
0000EFA8 E95FFAFFFF             <1>     jmp     m_pix_op_not_w_x
                                <1>
                                <1> m_pix_op_neg:
                                <1>     ; 06/02/2021
                                <1>     ; NEGATIVE COLOR (MASKED, full screen)
                                <1>     ;
                                <1>     ; jump from pix_op_neg
                                <1>     ;
                                <1>     ; INPUT:
                                <1>     ;   ecx = [v_siz] ; display page pixel count
                                <1>     ;   esi = edi = [v_mem] ; LFB start address
                                <1>     ;
                                <1>     ;   [maskcolor] = mask color (to be excluded)
                                <1>     ;
                                <1>     ; OUTPUT:
                                <1>     ;       [u.r0] will be > 0 if succesful
                                <1>
                                <1>     ; Full screen
                                <1> m_pix_op_neg_0:
0000EFAD 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000EFB4 7715                   <1>     ja      short m_pix_op_neg_1
                                <1>     ; 256 colors (8bpp)
                                <1>     ;jmp    short m_pix_op_neg_8
                                <1> m_pix_op_neg_8:
                                <1>     ; 8 bit colors (256 colors)
0000EFB6 AC                     <1>     lodsb
0000EFB7 3A05[6A9D0100]         <1>     cmp     al, [maskcolor]
0000EFBD 7408                   <1>     je      short m_pix_op_neg_8_1 ; exclude
0000EFBF F61F                   <1>     neg     byte [edi]
0000EFC1 FF05[348E0100]         <1>     inc     dword [u.r0] ; +1
                                <1> m_pix_op_neg_8_1:
0000EFC7 47                     <1>     inc     edi
0000EFC8 E2EC                   <1>     loop    m_pix_op_neg_8
0000EFCA C3                     <1>     retn
                                <1> m_pix_op_neg_1:
0000EFCB 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000EFD2 7746                   <1>     ja      short m_pix_op_neg_3 ; 32bpp
0000EFD4 7229                   <1>     jb      short m_pix_op_neg_2 ; 16bpp
                                <1>     ; 24 bit true colors
                                <1>     ;jmp    short m_pix_op_neg_24
                                <1> m_pix_op_neg_24:
                                <1>     ; 24 bit true colors
0000EFD6 66AD                   <1>     lodsw
0000EFD8 C1E010                 <1>     shl     eax, 16
0000EFDB AC                     <1>     lodsb
0000EFDC C1C010                 <1>     rol     eax, 16
0000EFDF 3B05[6A9D0100]         <1>     cmp     eax, [maskcolor]
0000EFE5 7412                   <1>     je      short m_pix_op_neg_24_1 ; exclude
0000EFE7 F7D8                   <1>     neg     eax
0000EFE9 668907                 <1>     mov     [edi], ax
```

```
9139 0000EFEC C1E810              <1>     shr     eax, 16
9140 0000EFEF 884702              <1>     mov     [edi+2], al
9141 0000EFF2 8305[348E0100]03    <1>     add     dword [u.r0], 3 ; +3
9142                              <1> m_pix_op_neg_24_1:
9143 0000EFF9 83C703              <1>     add     edi, 3 ; +3
9144 0000EFFC E2D8                <1>     loop    m_pix_op_neg_24
9145 0000EFFE C3                  <1>     retn
9146                              <1>     ; 65536 colors (16bpp)
9147                              <1> m_pix_op_neg_2:
9148                              <1>     ;jmp    short m_pix_op_neg_16
9149                              <1> m_pix_op_neg_16:
9150                              <1>     ; 16 bit colors (65536 colors)
9151 0000EFFF 66AD                <1>     lodsw
9152 0000F001 663B05[6A9D0100]    <1>     cmp     ax, [maskcolor]
9153 0000F008 740A                <1>     je      short m_pix_op_neg_16_1 ; exclude
9154 0000F00A 66F71F              <1>     neg     word [edi]
9155 0000F00D 8305[348E0100]02    <1>     add     dword [u.r0], 2 ; +2
9156                              <1> m_pix_op_neg_16_1:
9157 0000F014 83C702              <1>     add     edi, 2 ; +2
9158 0000F017 E2E6                <1>     loop    m_pix_op_neg_16
9159 0000F019 C3                  <1>     retn
9160                              <1> m_pix_op_neg_3:
9161                              <1>     ; 32 bit true colors
9162                              <1>     ;jmp    short m_pix_op_neg_32
9163                              <1> m_pix_op_neg_32:
9164                              <1>     ; 32 bit true colors
9165 0000F01A AD                  <1>     lodsd
9166 0000F01B 3B05[6A9D0100]      <1>     cmp     eax, [maskcolor]
9167 0000F021 7409                <1>     je      short m_pix_op_neg_32_1 ; exclude
9168 0000F023 F71F                <1>     neg     dword [edi]
9169 0000F025 8305[348E0100]04    <1>     add     dword [u.r0], 4 ; +4
9170                              <1> m_pix_op_neg_32_1:
9171 0000F02C 83C704              <1>     add     edi, 4 ; +4
9172 0000F02F E2E9                <1>     loop    m_pix_op_neg_32
9173 0000F031 C3                  <1>     retn
9174                              <1>
9175                              <1> m_pix_op_neg_w:
9176                              <1>     ; 06/02/2021
9177                              <1>     ; NEGATIVE COLOR (MASKED, window)
9178                              <1>     ;
9179                              <1>     ; jump from pix_op_neg_w
9180                              <1>     ;
9181                              <1>     ; INPUT:
9182                              <1>     ;   ecx = bytes per row (to be applied)
9183                              <1>     ;   edx = screen width in bytes
9184                              <1>     ;   ebx = row count
9185                              <1>     ;
9186                              <1>     ;   [maskcolor] = mask color (to be excluded)
9187                              <1>     ;
9188                              <1>     ; OUTPUT:
9189                              <1>     ;       [u.r0] will be > 0 if succesful
9190                              <1>
9191                              <1>     ; window
9192                              <1>     ;mov    edi, [v_str] ; LFB start address
9193                              <1>     ;mov    esi, edi
9194                              <1>
9195 0000F032 803D[599D0100]08    <1>     cmp     byte [v_bpp], 8 ; 8bpp
9196 0000F039 7707                <1>     ja      short m_pix_op_neg_w_1
9197                              <1>
9198                              <1>     ; 256 colors (8bpp)
9199 0000F03B BD[B6EF0000]        <1>     mov     ebp, m_pix_op_neg_8
9200 0000F040 EB1E                <1>     jmp     short m_pix_op_neg_w_4
9201                              <1>
9202                              <1> m_pix_op_neg_w_1:
9203 0000F042 803D[599D0100]18    <1>     cmp     byte [v_bpp], 24 ; 24bpp
9204 0000F049 7710                <1>     ja      short m_pix_op_neg_w_3 ; 32bpp
9205 0000F04B 7207                <1>     jb      short m_pix_op_neg_w_2 ; 16bpp
9206                              <1>
9207                              <1>     ; 24 bit true colors
9208 0000F04D BD[D6EF0000]        <1>     mov     ebp, m_pix_op_neg_24
9209 0000F052 EB0C                <1>     jmp     short m_pix_op_neg_w_4
9210                              <1>
9211                              <1>     ; 65536 colors (16bpp)
9212                              <1> m_pix_op_neg_w_2:
9213 0000F054 BD[FFEF0000]        <1>     mov     ebp, m_pix_op_neg_16
9214 0000F059 EB05                <1>     jmp     short m_pix_op_neg_w_4
9215                              <1>
9216                              <1>     ; 32 bit true colors
9217                              <1> m_pix_op_neg_w_3:
9218 0000F05B BD[1AF00000]        <1>     mov     ebp, m_pix_op_neg_32
9219                              <1> m_pix_op_neg_w_4:
9220 0000F060 E9A7F9FFFF          <1>     jmp     m_pix_op_neg_w_x
9221                              <1>
9222                              <1> m_pix_op_inc:
9223                              <1>     ; 06/02/2021
9224                              <1>     ; INCREASE COLOR (MASKED, full screen)
9225                              <1>     ;
9226                              <1>     ; jump from pix_op_inc
9227                              <1>     ;
9228                              <1>     ; INPUT:
9229                              <1>     ;   ecx = [v_siz] ; display page pixel count
9230                              <1>     ;   esi = edi = [v_mem] ; LFB start address
9231                              <1>     ;
9232                              <1>     ;   [maskcolor] = mask color (to be excluded)
9233                              <1>     ;
9234                              <1>     ; OUTPUT:
9235                              <1>     ;       [u.r0] will be > 0 if succesful
9236                              <1>
9237                              <1>     ; Full screen
9238                              <1> m_pix_op_inc_0:
9239 0000F065 803D[599D0100]08    <1>     cmp     byte [v_bpp], 8 ; 8bpp
9240 0000F06C 7719                <1>     ja      short m_pix_op_inc_1
9241                              <1>     ; 256 colors (8bpp)
9242                              <1>     ;jmp    short m_pix_op_inc_8
9243                              <1> m_pix_op_inc_8:
9244                              <1>     ; 8 bit colors (256 colors)
9245 0000F06E AC                  <1>     lodsb
9246 0000F06F 3A05[6A9D0100]      <1>     cmp     al, [maskcolor]
9247 0000F075 740C                <1>     je      short m_pix_op_inc_8_1 ; exclude
9248 0000F077 FE07                <1>     inc     byte [edi]
9249 0000F079 7502                <1>     jnz     short m_pix_op_inc_8_0
9250 0000F07B FE0F                <1>     dec     byte [edi]
9251                              <1> m_pix_op_inc_8_0:
9252 0000F07D FF05[348E0100]      <1>     inc     dword [u.r0] ; +1
9253                              <1> m_pix_op_inc_8_1:
9254 0000F083 47                  <1>     inc     edi
9255 0000F084 E2E8                <1>     loop    m_pix_op_inc_8
9256 0000F086 C3                  <1>     retn
9257                              <1> m_pix_op_inc_1:
9258 0000F087 803D[599D0100]18    <1>     cmp     byte [v_bpp], 24 ; 24bpp
9259 0000F08E 7752                <1>     ja      short m_pix_op_inc_3 ; 32bpp
9260 0000F090 7230                <1>     jb      short m_pix_op_inc_2 ; 16bpp
9261                              <1>     ; 24 bit true colors
9262                              <1>     ;jmp    short m_pix_op_inc_24
```

```
                                <1> m_pix_op_inc_24:
                                <1>     ; 24 bit true colors
0000F092 66AD                   <1>     lodsw
0000F094 C1E010                 <1>     shl     eax, 16
0000F097 AC                     <1>     lodsb
0000F098 C1C010                 <1>     rol     eax, 16
0000F09B 3B05[6A9D0100]         <1>     cmp     eax, [maskcolor]
0000F0A1 7419                   <1>     je      short m_pix_op_inc_24_1 ; exclude
0000F0A3 40                     <1>     inc     eax
0000F0A4 3DFFFFFF00             <1>     cmp     eax, 0FFFFFFh
0000F0A9 7601                   <1>     jna     short m_pix_op_inc_24_0
0000F0AB 48                     <1>     dec     eax
                                <1> m_pix_op_inc_24_0:
0000F0AC 668907                 <1>     mov     [edi], ax
0000F0AF C1E810                 <1>     shr     eax, 16
0000F0B2 884702                 <1>     mov     [edi+2], al
0000F0B5 8305[348E0100]03       <1>     add     dword [u.r0], 3 ; +3
                                <1> m_pix_op_inc_24_1:
0000F0BC 83C703                 <1>     add     edi, 3 ; +3
0000F0BF E2D1                   <1>     loop    m_pix_op_inc_24
0000F0C1 C3                     <1>     retn
                                <1>     ; 65536 colors (16bpp)
                                <1> m_pix_op_inc_2:
                                <1>     ;jmp    short m_pix_op_inc_16
                                <1> m_pix_op_inc_16:
                                <1>     ; 16 bit colors (65536 colors)
0000F0C2 66AD                   <1>     lodsw
0000F0C4 663B05[6A9D0100]       <1>     cmp     ax, [maskcolor]
0000F0CB 740F                   <1>     je      short m_pix_op_inc_16_1 ; exclude
0000F0CD 66FF07                 <1>     inc     word [edi]
0000F0D0 7503                   <1>     jnz     short m_pix_op_inc_16_0
0000F0D2 66FF0F                 <1>     dec     word [edi]
                                <1> m_pix_op_inc_16_0:
0000F0D5 8305[348E0100]02       <1>     add     dword [u.r0], 2 ; +2
                                <1> m_pix_op_inc_16_1:
0000F0DC 83C702                 <1>     add     edi, 2 ; +2
0000F0DF E2E1                   <1>     loop    m_pix_op_inc_16
0000F0E1 C3                     <1>     retn
                                <1> m_pix_op_inc_3:
                                <1>     ; 32 bit true colors
                                <1>     ;jmp    short m_pix_op_inc_32
                                <1> m_pix_op_inc_32:
                                <1>     ; 32 bit true colors
0000F0E2 AD                     <1>     lodsd
0000F0E3 3B05[6A9D0100]         <1>     cmp     eax, [maskcolor]
0000F0E9 740D                   <1>     je      short m_pix_op_inc_32_1 ; exclude
0000F0EB FF07                   <1>     inc     dword [edi]
0000F0ED 7502                   <1>     jnz     short m_pix_op_inc_32_0
0000F0EF FF0F                   <1>     dec     dword [edi]
                                <1> m_pix_op_inc_32_0:
0000F0F1 8305[348E0100]04       <1>     add     dword [u.r0], 4 ; +4
                                <1> m_pix_op_inc_32_1:
0000F0F8 83C704                 <1>     add     edi, 4 ; +4
0000F0FB E2E5                   <1>     loop    m_pix_op_inc_32
0000F0FD C3                     <1>     retn
                                <1>
                                <1> m_pix_op_inc_w:
                                <1>     ; 06/02/2021
                                <1>     ; INCREASE COLOR (MASKED, window)
                                <1>     ;
                                <1>     ; jump from pix_op_inc_w
                                <1>     ;
                                <1>     ; INPUT:
                                <1>     ;   ecx = bytes per row (to be applied)
                                <1>     ;   edx = screen width in bytes
                                <1>     ;   ebx = row count
                                <1>     ;
                                <1>     ;   [maskcolor] = mask color (to be excluded)
                                <1>     ;
                                <1>     ; OUTPUT:
                                <1>     ;       [u.r0] will be > 0 if succesful
                                <1>
                                <1>     ; window
                                <1>     ;mov    edi, [v_str] ; LFB start address
                                <1>     ;mov    esi, edi
                                <1>
0000F0FE 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000F105 7707                   <1>     ja      short m_pix_op_inc_w_1
                                <1>
                                <1>     ; 256 colors (8bpp)
0000F107 BD[6EF00000]           <1>     mov     ebp, m_pix_op_inc_8
0000F10C EB1E                   <1>     jmp     short m_pix_op_inc_w_4
                                <1>
                                <1> m_pix_op_inc_w_1:
0000F10E 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000F115 7710                   <1>     ja      short m_pix_op_inc_w_3 ; 32bpp
0000F117 7207                   <1>     jb      short m_pix_op_inc_w_2 ; 16bpp
                                <1>
0000F119 BD[92F00000]           <1>     ; 24 bit true colors
0000F11E EB0C                   <1>     mov     ebp, m_pix_op_inc_24
                                <1>     jmp     short m_pix_op_inc_w_4
                                <1>
0000F120 BD[C2F00000]           <1>     ; 65536 colors (16bpp)
0000F125 EB05                   <1> m_pix_op_inc_w_2:
                                <1>     mov     ebp, m_pix_op_inc_16
                                <1>     jmp     short m_pix_op_inc_w_4
                                <1>
0000F127 BD[E2F00000]           <1>     ; 32 bit true colors
                                <1> m_pix_op_inc_w_3:
0000F12C E9DBF8FFFF             <1>     mov     ebp, m_pix_op_inc_32
                                <1> m_pix_op_inc_w_4:
                                <1>     jmp     m_pix_op_inc_w_x
                                <1>
                                <1> m_pix_op_dec:
                                <1>     ; 06/02/2021
                                <1>     ; DECREASE COLOR (MASKED, full screen)
                                <1>     ;
                                <1>     ; jump from pix_op_dec
                                <1>     ;
                                <1>     ; INPUT:
                                <1>     ;   ecx = [v_siz] ; display page pixel count
                                <1>     ;   esi = edi = [v_mem] ; LFB start address
                                <1>     ;
                                <1>     ;   [maskcolor] = mask color (to be excluded)
                                <1>     ;
                                <1>     ; OUTPUT:
                                <1>     ;       [u.r0] will be > 0 if succesful
                                <1>
                                <1>     ; Full screen
                                <1> m_pix_op_dec_0:
0000F131 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000F138 7719                   <1>     ja      short m_pix_op_dec_1
                                <1>     ; 256 colors (8bpp)
                                <1>     ;jmp    short m_pix_op_dec_8
```

```
                                <1> m_pix_op_dec_8:
                                <1>     ; 8 bit colors (256 colors)
0000F13A AC                     <1>     lodsb
0000F13B 3A05[6A9D0100]         <1>     cmp     al, [maskcolor]
0000F141 740C                   <1>     je      short m_pix_op_dec_8_1 ; exclude
0000F143 FE0F                   <1>     dec     byte [edi]
0000F145 7902                   <1>     jns     short m_pix_op_dec_8_0
0000F147 FE07                   <1>     inc     byte [edi]
                                <1> m_pix_op_dec_8_0:
0000F149 FF05[348E0100]         <1>     inc     dword [u.r0] ; +1
                                <1> m_pix_op_dec_8_1:
0000F14F 47                     <1>     inc     edi
0000F150 E2E8                   <1>     loop    m_pix_op_dec_8
0000F152 C3                     <1>     retn
                                <1> m_pix_op_dec_1:
0000F153 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000F15A 774D                   <1>     ja      short m_pix_op_dec_3 ; 32bpp
0000F15C 722B                   <1>     jb      short m_pix_op_dec_2 ; 16bpp
                                <1>     ; 24 bit true colors
                                <1>     ;jmp    short m_pix_op_dec_24
                                <1> m_pix_op_dec_24:
                                <1>     ; 24 bit true colors
0000F15E 66AD                   <1>     lodsw
0000F160 C1E010                 <1>     shl     eax, 16
0000F163 AC                     <1>     lodsb
0000F164 C1C010                 <1>     rol     eax, 16
0000F167 3B05[6A9D0100]         <1>     cmp     eax, [maskcolor]
0000F16D 7414                   <1>     je      short m_pix_op_dec_24_1 ; exclude
0000F16F 48                     <1>     dec     eax
0000F170 7901                   <1>     jns     short m_pix_op_dec_24_0
0000F172 40                     <1>     inc     eax
                                <1> m_pix_op_dec_24_0:
0000F173 668907                 <1>     mov     [edi], ax
0000F176 C1E810                 <1>     shr     eax, 16
0000F179 884702                 <1>     mov     [edi+2], al
0000F17C 8305[348E0100]03       <1>     add     dword [u.r0], 3 ; +3
                                <1> m_pix_op_dec_24_1:
0000F183 83C703                 <1>     add     edi, 3 ; +3
0000F186 E2D6                   <1>     loop    m_pix_op_dec_24
0000F188 C3                     <1>     retn
                                <1>     ; 65536 colors (16bpp)
                                <1> m_pix_op_dec_2:
                                <1>     ;jmp    short m_pix_op_dec_16
                                <1> m_pix_op_dec_16:
                                <1>     ; 16 bit colors (65536 colors)
0000F189 66AD                   <1>     lodsw
0000F18B 663B05[6A9D0100]       <1>     cmp     ax, [maskcolor]
0000F192 740F                   <1>     je      short m_pix_op_dec_16_1 ; exclude
0000F194 66FF0F                 <1>     dec     word [edi]
0000F197 7903                   <1>     jns     short m_pix_op_dec_16_0
0000F199 66FF07                 <1>     inc     word [edi]
                                <1> m_pix_op_dec_16_0:
0000F19C 8305[348E0100]02       <1>     add     dword [u.r0], 2 ; +2
                                <1> m_pix_op_dec_16_1:
0000F1A3 83C702                 <1>     add     edi, 2 ; +2
0000F1A6 E2E1                   <1>     loop    m_pix_op_dec_16
0000F1A8 C3                     <1>     retn
                                <1> m_pix_op_dec_3:
                                <1>     ; 32 bit true colors
                                <1>     ;jmp    short m_pix_op_dec_32
                                <1> m_pix_op_dec_32:
                                <1>     ; 32 bit true colors
0000F1A9 AD                     <1>     lodsd
0000F1AA 3B05[6A9D0100]         <1>     cmp     eax, [maskcolor]
0000F1B0 740D                   <1>     je      short m_pix_op_dec_32_1 ; exclude
0000F1B2 FF0F                   <1>     dec     dword [edi]
0000F1B4 7902                   <1>     jns     short m_pix_op_dec_32_0
0000F1B6 FF07                   <1>     inc     dword [edi]
                                <1> m_pix_op_dec_32_0:
0000F1B8 8305[348E0100]04       <1>     add     dword [u.r0], 4 ; +4
                                <1> m_pix_op_dec_32_1:
0000F1BF 83C704                 <1>     add     edi, 4 ; +4
0000F1C2 E2E5                   <1>     loop    m_pix_op_dec_32
0000F1C4 C3                     <1>     retn
                                <1>
                                <1> m_pix_op_dec_w:
                                <1>     ; 06/02/2021
                                <1>     ; DECREASE COLOR (MASKED, window)
                                <1>     ;
                                <1>     ; jump from pix_op_dec_w
                                <1>     ;
                                <1>     ; INPUT:
                                <1>     ;   ecx = bytes per row (to be applied)
                                <1>     ;   edx = screen width in bytes
                                <1>     ;   ebx = row count
                                <1>     ;
                                <1>     ;   [maskcolor] = mask color (to be excluded)
                                <1>     ;
                                <1>     ; OUTPUT:
                                <1>     ;       [u.r0] will be > 0 if succesful
                                <1>
                                <1>     ; window
                                <1>     ;mov    edi, [v_str] ; LFB start address
                                <1>     ;mov    esi, edi
                                <1>
0000F1C5 803D[599D0100]08       <1>     cmp     byte [v_bpp], 8 ; 8bpp
0000F1CC 7707                   <1>     ja      short m_pix_op_dec_w_1
                                <1>
                                <1>     ; 256 colors (8bpp)
0000F1CE BD[3AF10000]           <1>     mov     ebp, m_pix_op_dec_8
0000F1D3 EB1E                   <1>     jmp     short m_pix_op_dec_w_4
                                <1>
                                <1> m_pix_op_dec_w_1:
0000F1D5 803D[599D0100]18       <1>     cmp     byte [v_bpp], 24 ; 24bpp
0000F1DC 7710                   <1>     ja      short m_pix_op_dec_w_3 ; 32bpp
0000F1DE 7207                   <1>     jb      short m_pix_op_dec_w_2 ; 16bpp
                                <1>
                                <1>     ; 24 bit true colors
0000F1E0 BD[5EF10000]           <1>     mov     ebp, m_pix_op_dec_24
0000F1E5 EB0C                   <1>     jmp     short m_pix_op_dec_w_4
                                <1>
                                <1>     ; 65536 colors (16bpp)
                                <1> m_pix_op_dec_w_2:
0000F1E7 BD[89F10000]           <1>     mov     ebp, m_pix_op_dec_16
0000F1EC EB05                   <1>     jmp     short m_pix_op_dec_w_4
                                <1>
                                <1>     ; 32 bit true colors
                                <1> m_pix_op_dec_w_3:
0000F1EE BD[A9F10000]           <1>     mov     ebp, m_pix_op_dec_32
                                <1> m_pix_op_dec_w_4:
0000F1F3 E914F8FFFF             <1>     jmp     m_pix_op_dec_w_x
                                <1>
                                <1> sysvideo_39:
                                <1>     ; 15/02/2021
```

```
9511                                <1>     ; 07/02/2021, 08/02/2021
9512                                <1>     ; 03/01/2021, 04/01/2021
9513                                <1>     ; 23/11/2020
9514                                <1>     ; BH = 3
9515                                <1>     ; PIXEL READ/WRITE
9516                                <1>
9517                                <1>     ; 07/02/2021
9518                                <1>     ; 04/01/2021 (TRDOS 386 v2.0.3)
9519 0000F1F8 80FB03               <1>     cmp     bl, 3
9520 0000F1FB 761A                 <1>     jna     short sysvideo_39_1
9521                                <1>     ; 07/02/2021
9522 0000F1FD 80FB06               <1>     cmp     bl, 6
9523 0000F200 7705                 <1>     ja      short sysvideo_39_0
9524 0000F202 E91A010000           <1>     jmp     sysvideo_39_31
9525                                <1> sysvideo_39_0:
9526                                <1>     ; error
9527 0000F207 B3FF                 <1>     mov     bl, 0FFh
9528 0000F209 8B2D[308E0100]       <1>     mov     ebp, [u.usp]  ; ebp points to user's registers
9529 0000F20F 895D10               <1>     mov     [ebp+16], ebx ; EBX
9530 0000F212 E967D8FFFF           <1>     jmp     sysret
9531                                <1> sysvideo_39_1:
9532 0000F217 803D[1E680000]FF     <1>     cmp     byte [CRT_MODE], 0FFh
9533 0000F21E 7312                 <1>     jnb     short sysvideo_39_2 ; SVGA (VESA VBE) video mode
9534                                <1>
9535                                <1>     ; Std VGA or CGA mode
9536 0000F220 81C200000A00         <1>     add     edx, 0A0000h
9537 0000F226 72DF                 <1>     jc      short sysvideo_39_0
9538 0000F228 81FAFFFF0A00         <1>     cmp     edx, 0AFFFFh
9539 0000F22E 77D7                 <1>     ja      short sysvideo_39_0
9540 0000F230 EB1E                 <1>     jmp     short sysvideo_39_3 ; 8bpp
9541                                <1>
9542                                <1> sysvideo_39_2:
9543                                <1>     ; use current vbe (svga) video mode
9544                                <1>
9545                                <1>     ; get LFB address
9546 0000F232 A1[FC9C0100]         <1>     mov     eax, [LFB_ADDR] ; [LFB_Info+LFBINFO.LFB_addr]
9547 0000F237 09C0                 <1>     or      eax, eax
9548 0000F239 74CC                 <1>     jz      short sysvideo_39_0
9549 0000F23B 3B15[009D0100]       <1>     cmp     edx, [LFB_SIZE]
9550 0000F241 73C4                 <1>     jnb     short sysvideo_39_0
9551                                <1>
9552 0000F243 01C2                 <1>     add     edx, eax
9553                                <1>     ;jc     short sysvideo_39_0
9554                                <1>
9555                                <1>     ; Pixel read/write in VESA VBE (2/3) video mode
9556                                <1>     ; Video memory at Linear Frame Buffer base address
9557                                <1>
9558 0000F245 8A3D[089D0100]       <1>     mov     bh, [LFB_Info+LFBINFO.bpp]
9559                                <1>
9560 0000F24B 80FF08               <1>     cmp     bh, 8 ; 8bpp
9561 0000F24E 775D                 <1>     ja      short sysvideo_39_17
9562                                <1>
9563                                <1>     ; 8 bits per pixel
9564                                <1> sysvideo_39_3:
9565 0000F250 80FB01               <1>     cmp     bl, 1 ; 1 = write pixel
9566 0000F253 7406                 <1>     je      short sysvideo_39_5
9567 0000F255 7712                 <1>     ja      short sysvideo_39_8
9568                                <1> sysvideo_39_4:
9569                                <1>     ; read pixel (8bpp)
9570 0000F257 8A02                 <1>     mov     al, [edx]
9571                                <1>     ;mov    [u.r0], al
9572                                <1>     ;jmp    sysret
9573 0000F259 EB04                 <1>     jmp     short sysvideo_39_7
9574                                <1> sysvideo_39_5:
9575                                <1>     ; write pixel (8bpp)
9576 0000F25B 88C8                 <1>     mov     al, cl
9577                                <1> sysvideo_39_6:
9578 0000F25D 8802                 <1>     mov     [edx], al
9579                                <1> sysvideo_39_7:
9580 0000F25F A2[348E0100]         <1>     mov     [u.r0], al
9581 0000F264 E915D8FFFF           <1>     jmp     sysret
9582                                <1> sysvideo_39_8:
9583 0000F269 80FB03               <1>     cmp     bl, 3 ; mix
9584 0000F26C 7208                 <1>     jb      short sysvideo_39_9
9585                                <1>     ; mix  pixel colors (8bpp)
9586 0000F26E 8A02                 <1>     mov     al, [edx]
9587 0000F270 00C8                 <1>     add     al, cl
9588 0000F272 D0D8                 <1>     rcr     al, 1
9589 0000F274 EBE7                 <1>     jmp     short sysvideo_39_6
9590                                <1> sysvideo_39_9:
9591                                <1>      ; swap pixel colors (8bpp)
9592 0000F276 88C8                 <1>     mov     al, cl
9593 0000F278 8602                 <1>     xchg    [edx], al
9594 0000F27A EBE3                 <1>     jmp     short sysvideo_39_7
9595                                <1>
9596                                <1>     ; 16 bits per pixel
9597                                <1> sysvideo_39_10:
9598 0000F27C 80FB01               <1>     cmp     bl, 1 ; 1 = write pixel
9599 0000F27F 7406                 <1>     je      short sysvideo_39_12
9600 0000F281 7714                 <1>     ja      short sysvideo_39_15
9601                                <1> sysvideo_39_11:
9602                                <1>     ; read pixel (16bpp)
9603 0000F283 8B02                 <1>     mov     eax, [edx]
9604                                <1>     ;mov    [u.r0], ax
9605                                <1>     ;jmp    sysret
9606 0000F285 EB05                 <1>     jmp     short sysvideo_39_14
9607                                <1> sysvideo_39_12:
9608                                <1>     ; write pixel (16bpp)
9609 0000F287 89C8                 <1>     mov     eax, ecx
9610                                <1> sysvideo_39_13:
9611 0000F289 668902               <1>     mov     [edx], ax
9612                                <1> sysvideo_39_14:
9613 0000F28C 66A3[348E0100]       <1>     mov     [u.r0], ax
9614 0000F292 E9E7D7FFFF           <1>     jmp     sysret
9615                                <1> sysvideo_39_15:
9616 0000F297 80FB03               <1>     cmp     bl, 3 ; mix
9617 0000F29A 720A                 <1>     jb      short sysvideo_39_16
9618                                <1>     ; mix  pixel colors (16bpp)
9619 0000F29C 8B02                 <1>     mov     eax, [edx]
9620 0000F29E 6601C8               <1>     add     ax, cx
9621 0000F2A1 66D1D8               <1>     rcr     ax, 1
9622 0000F2A4 EBE3                 <1>     jmp     short sysvideo_39_13
9623                                <1> sysvideo_39_16:
9624                                <1>      ; swap pixel colors (16bpp)
9625 0000F2A6 89C8                 <1>     mov     eax, ecx
9626 0000F2A8 668702               <1>     xchg    [edx], ax
9627 0000F2AB EBDF                 <1>     jmp     short sysvideo_39_14
9628                                <1> sysvideo_39_17:
9629 0000F2AD 80FF18               <1>     cmp     bh, 24
9630 0000F2B0 7743                 <1>     ja      short sysvideo_39_24
9631 0000F2B2 72C8                 <1>     jb      short sysvideo_39_10
9632                                <1>
9633                                <1>     ; 24 bits per pixel
9634 0000F2B4 81E1FFFFFF00         <1>     and     ecx, 0FFFFFFh
```

```
9635 0000F2BA 80FB01              <1>     cmp     bl, 1 ; 1 = write pixel
9636 0000F2BD 7406                <1>     je      short sysvideo_39_19
9637 0000F2BF 7712                <1>     ja      short sysvideo_39_22
9638                              <1> sysvideo_39_18:
9639                              <1>     ; read pixel (24bpp)
9640 0000F2C1 8B02                <1>     mov     eax, [edx]
9641                              <1>     ;and    eax, 0FFFFFFh
9642                              <1>     ;mov    [u.r0], eax
9643                              <1>     ;jmp    sysret
9644 0000F2C3 EB04                <1>     jmp     short sysvideo_39_21
9645                              <1> sysvideo_39_19:
9646                              <1>     ; write pixel (24bpp)
9647 0000F2C5 89C8                <1>     mov     eax, ecx
9648                              <1> sysvideo_39_20:
9649                              <1>     ;and    eax, 0FFFFFFh
9650 0000F2C7 8902                <1>     mov     [edx], eax
9651                              <1> sysvideo_39_21:
9652 0000F2C9 A3[348E0100]        <1>     mov     [u.r0], eax
9653 0000F2CE E9ABD7FFFF          <1>     jmp     sysret
9654                              <1> sysvideo_39_22:
9655 0000F2D3 80FB03              <1>     cmp     bl, 3 ; mix
9656 0000F2D6 720D                <1>     jb      short sysvideo_39_23
9657                              <1>     ; mix  pixel colors (24bpp)
9658 0000F2D8 8B02                <1>     mov     eax, [edx]
9659 0000F2DA 25FFFFFF00          <1>     and     eax, 0FFFFFFh
9660                              <1>     ;and    ecx, 0FFFFFFh
9661 0000F2DF 01C8                <1>     add     eax, ecx
9662 0000F2E1 D1D8                <1>     rcr     eax, 1
9663 0000F2E3 EBE2                <1>     jmp     short sysvideo_39_20
9664                              <1> sysvideo_39_23:
9665                              <1>      ; swap pixel colors (24bpp)
9666 0000F2E5 89C8                <1>     mov     eax, ecx
9667                              <1>     ;and    eax, 0FFFFFFh
9668 0000F2E7 668702              <1>     xchg    [edx], ax
9669 0000F2EA C1C810              <1>     ror     eax, 16
9670 0000F2ED 884202              <1>     mov     [edx+2], al
9671 0000F2F0 C1C010              <1>     rol     eax, 16
9672 0000F2F3 EBD4                <1>     jmp     short sysvideo_39_21
9673                              <1>
9674                              <1>     ; 32 bits per pixel
9675                              <1> sysvideo_39_24:
9676 0000F2F5 80FB01              <1>     cmp     bl, 1 ; 1 = write pixel
9677 0000F2F8 7406                <1>     je      short sysvideo_39_26
9678 0000F2FA 7712                <1>     ja      short sysvideo_39_29
9679                              <1> sysvideo_39_25:
9680                              <1>     ; read pixel (32bpp)
9681 0000F2FC 8B02                <1>     mov     eax, [edx]
9682                              <1>     ;mov    [u.r0], eax
9683                              <1>     ;jmp    sysret
9684 0000F2FE EB04                <1>     jmp     short sysvideo_39_28
9685                              <1> sysvideo_39_26:
9686                              <1>     ; write pixel (32bpp)
9687 0000F300 89C8                <1>     mov     eax, ecx
9688                              <1> sysvideo_39_27:
9689 0000F302 8902                <1>     mov     [edx], eax
9690                              <1> sysvideo_39_28:
9691 0000F304 A3[348E0100]        <1>     mov     [u.r0], eax
9692 0000F309 E970D7FFFF          <1>     jmp     sysret
9693                              <1> sysvideo_39_29:
9694 0000F30E 80FB03              <1>     cmp     bl, 3 ; mix
9695 0000F311 7208                <1>     jb      short sysvideo_39_30
9696                              <1>     ; mix  pixel colors (32bpp)
9697 0000F313 8B02                <1>     mov     eax, [edx]
9698 0000F315 01C8                <1>     add     eax, ecx
9699 0000F317 D1D8                <1>     rcr     eax, 1
9700 0000F319 EBE7                <1>     jmp     short sysvideo_39_27
9701                              <1> sysvideo_39_30:
9702                              <1>      ; swap pixel colors (32bpp)
9703 0000F31B 89C8                <1>     mov     eax, ecx
9704 0000F31D 8702                <1>     xchg    [edx], eax
9705 0000F31F EBE3                <1>     jmp     short sysvideo_39_28
9706                              <1>
9707                              <1> sysvideo_39_31:
9708                              <1>     ; 06/03/2021
9709                              <1>     ; 08/02/2021
9710                              <1>     ; 07/02/2021
9711                              <1>     ; BL = 4 -> read pixels from user defined positions
9712                              <1>     ; BL = 5 -> write single color pixels to user defined pos.
9713                              <1>     ; BL = 6 -> write multi color pixels to user defined pos.
9714                              <1>     ; ECX = color (CL, CX, ECX)
9715                              <1>     ; EDX = number of pixels
9716                              <1>     ; ESI = user buffer contains dword pixel positions
9717                              <1>     ;       (and dword colors for BL input = 6)
9718                              <1>     ; EDI = user's pixel color buff (destination) for BL = 4
9719                              <1>
9720 0000F321 890D[6A9D0100]      <1>     mov     [maskcolor], ecx
9721 0000F327 89D5                <1>     mov     ebp, edx ; number of pixels
9722 0000F329 803D[1E680000]FF    <1>     cmp     byte [CRT_MODE], 0FFh ; SVGA flag
9723 0000F330 7317                <1>     jnb     short sysvideo_39_33 ; SVGA (VESA VBE mode)
9724                              <1>     ; Standard VGA mode
9725 0000F332 B900000100          <1>     mov     ecx, 65536 ; Video page size (maximum)
9726 0000F337 39CA                <1>     cmp     edx, ecx
9727 0000F339 7709                <1>     ja      short sysvideo_39_32 ; abnormal value !
9728 0000F33B B800000A00          <1>     mov     eax, 0A0000h ; Video page start address
9729 0000F340 B708                <1>     mov     bh, 8  ; 8 bits per pixel (256 colors)
9730 0000F342 EB35                <1>     jmp     short sysvideo_39_34
9731                              <1> sysvideo_39_32:
9732                              <1>     ; nonsense! (edx has abnormal value)
9733 0000F344 E935D7FFFF          <1>     jmp     sysret
9734                              <1> sysvideo_39_33:
9735                              <1>     ; 06/03/2021
9736 0000F349 8A3D[089D0100]      <1>     mov     bh, [LFB_Info+LFBINFO.bpp]
9737 0000F34F 80FF08              <1>     cmp     bh, 8
9738 0000F352 7412                <1>     je      short sysvideo_39_81 ; 8bpp
9739 0000F354 89D0                <1>     mov     eax, edx
9740 0000F356 80FF10              <1>     cmp     bh, 16
9741 0000F359 7409                <1>     je      short sysvideo_39_80 ; 16bpp
9742 0000F35B D1E2                <1>     shl     edx, 1
9743 0000F35D 80FF20              <1>     cmp     bh, 32
9744 0000F360 7502                <1>     jne     short sysvideo_39_80 ; 24bpp
9745 0000F362 D1E0                <1>     shl     eax, 1
9746                              <1> sysvideo_39_80:
9747 0000F364 01C2                <1>     add     edx, eax
9748                              <1>     ; edx = number of bytes
9749                              <1> sysvideo_39_81:
9750                              <1>     ; get LFB address
9751 0000F366 A1[FC9C0100]        <1>     mov     eax, [LFB_ADDR] ; [LFB_Info+LFBINFO.LFB_addr]
9752 0000F36B 09C0                <1>     or      eax, eax
9753 0000F36D 74D5                <1>     jz      short sysvideo_39_32 ; LFB is not ready !
9754 0000F36F 8B0D[009D0100]      <1>     mov     ecx, [LFB_SIZE]
9755 0000F375 39CA                <1>     cmp     edx, ecx
9756 0000F377 77CB                <1>     ja      short sysvideo_39_32 ; abnormal value !
9757                              <1>
9758                              <1>     ; 02/03/2021
```

```
9759                                    <1>     ; 08/02/2021
9760                                    <1>     ;mov    ebp, edx ; pixel count
9761                                    <1>     ;shl    ebp, 2 ; byte count (pixel pos: 4 bytes)
9762                                    <1>
9763                                    <1>     ; 06/03/2021
9764                                    <1>     ; bits per pixel (pixel color size)
9765                                    <1>     ;mov    bh, [LFB_Info+LFBINFO.bpp]
9766                                    <1> sysvideo_39_34:
9767 0000F379 C1E502                   <1>     shl     ebp, 2 ; 15/02/2021 (byte count)
9768 0000F37C A3[5A9D0100]             <1>     mov     [v_mem], eax ; Save video page start address
9769 0000F381 88DE                     <1>     mov     dh, bl ; sub function
9770                                    <1>     ; 06/03/2021
9771 0000F383 883D[599D0100]           <1>     mov     [v_bpp], bh ; bits per pixel (color size)
9772                                    <1>     ;mov    ebx, [LFB_SIZE]
9773 0000F389 89CB                     <1>     mov     ebx, ecx ; [LFB_SIZE]
9774                                    <1>
9775 0000F38B B900080000               <1>     mov     ecx, 2048
9776 0000F390 39CD                     <1>     cmp     ebp, ecx
9777 0000F392 7302                     <1>     jnb     short sysvideo_39_35
9778 0000F394 89E9                     <1>     mov     ecx, ebp ; fix to requested byte count
9779                                    <1> sysvideo_39_35:
9780 0000F396 80FE04                   <1>     cmp     dh, 4 ; 08/02/2021
9781                                    <1>     ;cmp    bl, 4 ; read pixels from user defined positions
9782 0000F399 7605                     <1>     jna     short sysvideo_39_36
9783 0000F39B E9B2000000               <1>     jmp     sysvideo_39_52
9784                                    <1>     ; 08/02/2021
9785                                    <1>     ;mov    [buffer8], edi  ; user's destination buff addr
9786                                    <1> sysvideo_39_36:
9787                                    <1>     ; 08/02/2021
9788                                    <1>     ; read pixel positions
9789                                    <1>     ; as defined in user's source buffer
9790 0000F3A0 893D[729D0100]           <1>     mov     [buffer8], edi  ; user's destination buff addr
9791 0000F3A6 BF00760900               <1>     mov     edi, VBE3SAVERESTOREBLOCK ; kernel buffer for
9792                                    <1>                                       ; 2028 byte data
9793                                    <1>     ; esi = user's source buffer for pixel positions
9794                                    <1>     ; ecx = byte count
9795 0000F3AB E884170000               <1>     call    transfer_from_user_buffer
9796 0000F3B0 7292                     <1>     jc      short sysvideo_39_32 ; error
9797                                    <1>     ; ecx  = transfer count (bytes)
9798                                    <1>
9799 0000F3B2 57                       <1>     push    edi ; *
9800 0000F3B3 56                       <1>     push    esi ; **
9801 0000F3B4 51                       <1>     push    ecx ; ***
9802                                    <1>
9803 0000F3B5 89FE                     <1>     mov     esi, edi ; kernel buffer
9804 0000F3B7 8B15[5A9D0100]           <1>     mov     edx, [v_mem] ; video memory
9805 0000F3BD C1E902                   <1>     shr     ecx, 2 ; pixel count (within buffer capacity)
9806                                    <1>
9807 0000F3C0 803D[599D0100]08         <1>     cmp     byte [v_bpp], 8 ; 8bpp
9808 0000F3C7 7753                     <1>     ja      short sysvideo_39_49
9809                                    <1> sysvideo_39_37:
9810                                    <1>     ; 8bpp
9811 0000F3C9 AD                       <1>     lodsd
9812 0000F3CA 39D8                     <1>     cmp     eax, ebx ; < [LFB_SIZE]
9813 0000F3CC 7309                     <1>     jnb     short sysvideo_39_39
9814 0000F3CE 0FB60402                 <1>     movzx   eax, byte [edx+eax]
9815                                    <1> sysvideo_39_38:
9816 0000F3D2 AB                       <1>     stosd
9817 0000F3D3 E2F4                     <1>     loop    sysvideo_39_37
9818 0000F3D5 EB49                     <1>     jmp     short sysvideo_39_50
9819                                    <1> sysvideo_39_39:
9820                                    <1>     ; write black color for improper positions
9821 0000F3D7 31C0                     <1>     xor     eax, eax
9822 0000F3D9 EBF7                     <1>     jmp     short sysvideo_39_38
9823                                    <1> sysvideo_39_40:
9824 0000F3DB 803D[599D0100]18         <1>     cmp     byte [v_bpp], 24 ; 24bpp
9825 0000F3E2 772A                     <1>     ja      short sysvideo_39_47 ; 32bpp
9826 0000F3E4 7216                     <1>     jb      short sysvideo_39_44 ; 16bpp
9827                                    <1> sysvideo_39_41:
9828                                    <1>     ; 24bpp
9829 0000F3E6 AD                       <1>     lodsd
9830 0000F3E7 39D8                     <1>     cmp     eax, ebx ; < [LFB_SIZE]
9831 0000F3E9 730D                     <1>     jnb     short sysvideo_39_43
9832 0000F3EB 8B0402                   <1>     mov     eax, [edx+eax]
9833 0000F3EE 25FFFFFF00               <1>     and     eax, 0FFFFFFh
9834                                    <1> sysvideo_39_42:
9835 0000F3F3 AB                       <1>     stosd
9836 0000F3F4 E2F0                     <1>     loop    sysvideo_39_41
9837 0000F3F6 EB28                     <1>     jmp     short sysvideo_39_50
9838                                    <1> sysvideo_39_43:
9839                                    <1>     ; write black color for improper positions
9840 0000F3F8 31C0                     <1>     xor     eax, eax
9841 0000F3FA EBF7                     <1>     jmp     short sysvideo_39_42
9842                                    <1> sysvideo_39_44:
9843                                    <1>     ; 16bpp
9844 0000F3FC AD                       <1>     lodsd
9845 0000F3FD 39D8                     <1>     cmp     eax, ebx ; < [LFB_SIZE]
9846 0000F3FF 7309                     <1>     jnb     short sysvideo_39_46
9847 0000F401 0FB70402                 <1>     movzx   eax, word [edx+eax]
9848                                    <1> sysvideo_39_45:
9849 0000F405 AB                       <1>     stosd
9850 0000F406 E2F4                     <1>     loop    sysvideo_39_44
9851 0000F408 EB16                     <1>     jmp     short sysvideo_39_50
9852                                    <1> sysvideo_39_46:
9853                                    <1>     ; write black color for improper positions
9854 0000F40A 31C0                     <1>     xor     eax, eax
9855 0000F40C EBF7                     <1>     jmp     short sysvideo_39_45
9856                                    <1> sysvideo_39_47:
9857                                    <1>     ; 32bpp
9858 0000F40E AD                       <1>     lodsd
9859 0000F40F 39D8                     <1>     cmp     eax, ebx ; < [LFB_SIZE]
9860 0000F411 7309                     <1>     jnb     short sysvideo_39_49
9861 0000F413 0FB70402                 <1>     movzx   eax, word [edx+eax]
9862                                    <1> sysvideo_39_48:
9863 0000F417 AB                       <1>     stosd
9864 0000F418 E2F4                     <1>     loop    sysvideo_39_47
9865 0000F41A EB04                     <1>     jmp     short sysvideo_39_50
9866                                    <1> sysvideo_39_49:
9867                                    <1>     ; write black color for improper positions
9868 0000F41C 31C0                     <1>     xor     eax, eax
9869 0000F41E EBF7                     <1>     jmp     short sysvideo_39_48
9870                                    <1> sysvideo_39_50:
9871 0000F420 59                       <1>     pop     ecx ; transfer count in bytes
9872 0000F421 5E                       <1>     pop     esi ; ** ; kernel buffer
9873                                    <1>     ;mov    esi, VBE3SAVERESTOREBLOCK ; kernel buffer for
9874                                    <1>                                       ; 2048 byte data
9875 0000F422 8B3D[729D0100]           <1>     mov     edi, [buffer8]
9876                                    <1>     ; edi = user's destination buffer for pixel colors
9877                                    <1>     ; ecx = byte count
9878 0000F428 E8BD160000               <1>     call    transfer_to_user_buffer
9879 0000F42D 5E                       <1>     pop     esi ; *
9880 0000F42E 7266                     <1>     jc      short sysvideo_39_56 ; error
9881                                    <1>     ; ecx  = transfer count (bytes)
9882 0000F430 89C8                     <1>     mov     eax, ecx
```

```
9883 0000F432 C1E802              <1>     shr     eax, 2
9884 0000F435 0105[348E0100]      <1>     add     [u.r0], eax ; transfer count (in pixels)
9885                              <1>
9886 0000F43B 29CD                <1>     sub     ebp, ecx
9887 0000F43D 7657                <1>     jna     short sysvideo_39_56 ; completed/finished
9888 0000F43F 01CE                <1>     add     esi, ecx ; next position in source buffer
9889                              <1>     ;add    [buffer8], ecx ; next pos in destination buff
9890 0000F441 01CF                <1>     add     edi, ecx
9891 0000F443 66B90008            <1>     mov     cx, 2048 ; new count, limit: kernel buff size
9892 0000F447 39CD                <1>     cmp     ebp, ecx ; remain >= limit ?
9893 0000F449 7302                <1>     jnb     short sysvideo_39_51 ; yes
9894 0000F44B 89E9                <1>     mov     ecx, ebp ; fix byte count to remain bytes
9895                              <1> sysvideo_39_51:
9896 0000F44D E94EFFFFFF          <1>     jmp     sysvideo_39_36
9897                              <1>
9898                              <1> sysvideo_39_52:
9899 0000F452 80FE05              <1>     cmp     dh, 5 ; 08/02/2021
9900                              <1>     ;cmp    bl, 5 ; write pixels to user defined positions
9901 0000F455 7605                <1>     jna     short sysvideo_39_53
9902 0000F457 E9A1000000          <1>     jmp     sysvideo_39_66
9903                              <1> sysvideo_39_53:
9904                              <1>     ; single color pixel writing
9905 0000F45C BF00760900          <1>     mov     edi, VBE3SAVERESTOREBLOCK ; kernel buffer for
9906                              <1>                                       ; 2028 byte data
9907                              <1>     ; esi = user's source buffer for pixel positions
9908                              <1>     ; ecx = byte count
9909 0000F461 E8CE160000          <1>     call    transfer_from_user_buffer
9910 0000F466 722E                <1>     jc      short sysvideo_39_56 ; error
9911                              <1>     ; ecx = transfer count (bytes)
9912                              <1>
9913                              <1>     ; write pixels by using (user) defined positions
9914                              <1>     ; ecx = byte count (1,2,3,4 times pixel count)
9915                              <1>     ; edi = system buffer address
9916                              <1>
9917 0000F468 56                  <1>     push    esi ; *
9918 0000F469 51                  <1>     push    ecx ; **
9919                              <1>
9920 0000F46A 89FE                <1>     mov     esi, edi
9921 0000F46C 8B3D[5A9D0100]      <1>     mov     edi, [v_mem]
9922                              <1>
9923                              <1>     ; 08/02/2021
9924 0000F472 C1E902              <1>     shr     ecx, 2 ; pixel count
9925 0000F475 8B15[6A9D0100]      <1>     mov     edx, [maskcolor]
9926                              <1>     ;mov    ebx, [v_siz]
9927                              <1>
9928 0000F47B 803D[599D0100]08    <1>     cmp     byte [v_bpp], 8 ; 8bpp
9929 0000F482 7717                <1>     ja      short sysvideo_39_57
9930                              <1> sysvideo_39_54:
9931                              <1>     ; 8bpp
9932 0000F484 AD                  <1>     lodsd
9933 0000F485 39D8                <1>     cmp     eax, ebx ; < [v_siz]
9934 0000F487 7309                <1>     jnb     short sysvideo_39_55
9935 0000F489 881407              <1>     mov     [edi+eax], dl
9936                              <1>     ; 06/03/2021
9937 0000F48C FF05[348E0100]      <1>     inc     dword [u.r0]
9938                              <1> sysvideo_39_55:
9939 0000F492 E2F0                <1>     loop    sysvideo_39_54
9940 0000F494 EB50                <1>     jmp     short sysvideo_39_64
9941                              <1> sysvideo_39_56:
9942 0000F496 E9E3D5FFFF          <1>     jmp     sysret
9943                              <1> sysvideo_39_57:
9944 0000F49B 803D[599D0100]18    <1>     cmp     byte [v_bpp], 24 ; 24bpp
9945 0000F4A2 7732                <1>     ja      short sysvideo_39_62 ; 32bpp
9946 0000F4A4 721D                <1>     jb      short sysvideo_39_60 ; 16bpp
9947                              <1> sysvideo_39_58:
9948                              <1>     ; 24bpp
9949 0000F4A6 AD                  <1>     lodsd
9950 0000F4A7 39D8                <1>     cmp     eax, ebx ; < [v_siz]
9951 0000F4A9 7314                <1>     jnb     short sysvideo_39_59
9952 0000F4AB 881407              <1>     mov     [edi+eax], dl
9953 0000F4AE 40                  <1>     inc     eax
9954 0000F4AF C1CA08              <1>     ror     edx, 8
9955 0000F4B2 66891407            <1>     mov     [edi+eax], dx
9956 0000F4B6 C1C208              <1>     rol     edx, 8
9957 0000F4B9 FF05[348E0100]      <1>     inc     dword [u.r0]
9958                              <1> sysvideo_39_59:
9959 0000F4BF E2E5                <1>     loop    sysvideo_39_58
9960 0000F4C1 EB23                <1>     jmp     short sysvideo_39_64
9961                              <1> sysvideo_39_60:
9962                              <1>     ; 16bpp
9963 0000F4C3 AD                  <1>     lodsd
9964 0000F4C4 39D8                <1>     cmp     eax, ebx ; < [v_siz]
9965 0000F4C6 730A                <1>     jnb     short sysvideo_39_61
9966 0000F4C8 66891407            <1>     mov     [edi+eax], dx
9967 0000F4CC FF05[348E0100]      <1>     inc     dword [u.r0]
9968                              <1> sysvideo_39_61:
9969 0000F4D2 E2EF                <1>     loop    sysvideo_39_60
9970 0000F4D4 EB10                <1>     jmp     short sysvideo_39_64
9971                              <1> sysvideo_39_62:
9972                              <1>     ; 32bpp
9973 0000F4D6 AD                  <1>     lodsd
9974 0000F4D7 39D8                <1>     cmp     eax, ebx ; < [v_siz]
9975 0000F4D9 7309                <1>     jnb     short sysvideo_39_63
9976 0000F4DB 891407              <1>     mov     [edi+eax], edx
9977 0000F4DE FF05[348E0100]      <1>     inc     dword [u.r0]
9978                              <1> sysvideo_39_63:
9979 0000F4E4 E2F0                <1>     loop    sysvideo_39_62
9980                              <1> sysvideo_39_64:
9981 0000F4E6 59                  <1>     pop     ecx ; **
9982 0000F4E7 5E                  <1>     pop     esi ; *
9983 0000F4E8 29CD                <1>     sub     ebp, ecx
9984 0000F4EA 76AA                <1>     jna     short sysvideo_39_56
9985 0000F4EC 01CE                <1>     add     esi, ecx
9986 0000F4EE 66B90008            <1>     mov     cx, 2048
9987 0000F4F2 39CD                <1>     cmp     ebp, ecx
9988 0000F4F4 7302                <1>     jnb     short sysvideo_39_65
9989 0000F4F6 89E9                <1>     mov     ecx, ebp
9990                              <1> sysvideo_39_65:
9991 0000F4F8 E95FFFFFFF          <1>     jmp     sysvideo_39_53
9992                              <1>
9993                              <1> sysvideo_39_66:
9994                              <1>     ; 15/02/2021
9995 0000F4FD D1E5                <1>     shl     ebp, 1 ; 8 bytes per pixel (position&color)
9996                              <1> sysvideo_39_67:
9997 0000F4FF 66B90008            <1>     mov     cx, 2048
9998 0000F503 39CD                <1>     cmp     ebp, ecx
9999 0000F505 7302                <1>     jnb     short sysvideo_39_68
10000 0000F507 89E9               <1>     mov     ecx, ebp
10001                             <1> sysvideo_39_68:
10002                             <1>     ; multi colors pixel writing
10003 0000F509 BF00760900         <1>     mov     edi, VBE3SAVERESTOREBLOCK ; kernel buffer for
10004                             <1>                                       ; 2048 byte data
10005                             <1>     ; esi = user's source buffer for pixel positions
10006                             <1>     ; ecx = byte count
```

```
10007 0000F50E E821160000          <1>     call    transfer_from_user_buffer
10008 0000F513 7281                <1>     jc      short sysvideo_39_56 ; error
10009                              <1>     ; ecx  = transfer count
10010                              <1>
10011                              <1>     ; write pixels & colors as defined in user buffer
10012                              <1>     ; ecx = byte count (2,4,6,8 times pixel count)
10013                              <1>     ; edi = system buffer address
10014                              <1>
10015 0000F515 56                  <1>     push    esi ; **
10016 0000F516 51                  <1>     push    ecx ; *
10017                              <1>
10018 0000F517 89FE                <1>     mov     esi, edi
10019 0000F519 8B3D[5A9D0100]      <1>     mov     edi, [v_mem]
10020                              <1>
10021                              <1>     ; 08/02/2021
10022 0000F51F C1E903              <1>     shr     ecx, 3 ; pixel count
10023                              <1>
10024                              <1>     ;mov    ebx, [v_siz]
10025                              <1>
10026 0000F522 803D[599D0100]08    <1>     cmp     byte [v_bpp], 8 ; 8bpp
10027 0000F529 7715                <1>     ja      short sysvideo_39_71
10028                              <1> sysvideo_39_69:
10029                              <1>     ; 8bpp
10030 0000F52B AD                  <1>     lodsd   ; position
10031 0000F52C 89C2                <1>     mov     edx, eax
10032 0000F52E AD                  <1>     lodsd   ; color
10033 0000F52F 39DA                <1>     cmp     edx, ebx ; < [v_siz]
10034 0000F531 7309                <1>     jnb     short sysvideo_39_70
10035 0000F533 880417              <1>     mov     [edi+edx], al
10036                              <1>     ; 06/03/2021
10037 0000F536 FF05[348E0100]      <1>     inc     dword [u.r0]
10038                              <1> sysvideo_39_70:
10039 0000F53C E2ED                <1>     loop    sysvideo_39_69
10040 0000F53E EB51                <1>     jmp     short sysvideo_39_78
10041                              <1> sysvideo_39_71:
10042 0000F540 803D[599D0100]18    <1>     cmp     byte [v_bpp], 24 ; 24bpp
10043 0000F547 7735                <1>     ja      short sysvideo_39_76 ; 32bpp
10044 0000F549 721D                <1>     jb      short sysvideo_39_74 ; 16bpp
10045                              <1> sysvideo_39_72:
10046                              <1>     ; 24bpp
10047 0000F54B AD                  <1>     lodsd   ; position
10048 0000F54C 89C2                <1>     mov     edx, eax
10049 0000F54E AD                  <1>     lodsd   ; color
10050 0000F54F 39DA                <1>     cmp     edx, ebx ; < [v_siz]
10051 0000F551 7311                <1>     jnb     short sysvideo_39_73
10052 0000F553 880417              <1>     mov     [edi+edx], al
10053 0000F556 42                  <1>     inc     edx
10054 0000F557 C1E808              <1>     shr     eax, 8
10055 0000F55A 66890417            <1>     mov     [edi+edx], ax
10056 0000F55E FF05[348E0100]      <1>     inc     dword [u.r0]
10057                              <1> sysvideo_39_73:
10058 0000F564 E2E5                <1>     loop    sysvideo_39_72
10059 0000F566 EB29                <1>     jmp     short sysvideo_39_78
10060                              <1> sysvideo_39_74:
10061                              <1>     ; 16bpp
10062 0000F568 AD                  <1>     lodsd   ; position
10063 0000F569 89C2                <1>     mov     edx, eax
10064 0000F56B AD                  <1>     lodsd   ; color
10065 0000F56C 39DA                <1>     cmp     edx, ebx ; < [v_siz]
10066 0000F56E 730A                <1>     jnb     short sysvideo_39_75
10067 0000F570 66890417            <1>     mov     [edi+edx], ax
10068 0000F574 FF05[348E0100]      <1>     inc     dword [u.r0]
10069                              <1> sysvideo_39_75:
10070 0000F57A E2EC                <1>     loop    sysvideo_39_74
10071 0000F57C EB13                <1>     jmp     short sysvideo_39_78
10072                              <1> sysvideo_39_76:
10073                              <1>     ; 32bpp
10074 0000F57E AD                  <1>     lodsd   ; position
10075 0000F57F 89C2                <1>     mov     edx, eax
10076 0000F581 AD                  <1>     lodsd   ; color
10077 0000F582 39DA                <1>     cmp     edx, ebx ; < [v_siz]
10078 0000F584 7309                <1>     jnb     short sysvideo_39_77
10079 0000F586 890417              <1>     mov     [edi+edx], eax
10080 0000F589 FF05[348E0100]      <1>     inc     dword [u.r0]
10081                              <1> sysvideo_39_77:
10082 0000F58F E2ED                <1>     loop    sysvideo_39_76
10083                              <1> sysvideo_39_78:
10084 0000F591 59                  <1>     pop     ecx ; *
10085 0000F592 5E                  <1>     pop     esi ; **
10086                              <1>
10087 0000F593 29CD                <1>     sub     ebp, ecx
10088 0000F595 762A                <1>     jna     short sysvideo_39_79
10089 0000F597 01CE                <1>     add     esi, ecx
10090 0000F599 E961FFFFFF          <1>     jmp     sysvideo_39_67
10091                              <1> ;sysvideo_39_79:
10092                              <1> ;   jmp     sysret
10093                              <1>
10094                              <1> sysvideo_16:
10095                              <1>     ; 11/08/2022
10096                              <1>     ; 23/07/2022
10097                              <1>     ; 06/03/2021
10098                              <1>     ; 23/11/2020
10099 0000F59E 80FF04              <1>     cmp     bh, 4
10100                              <1>     ;jb     sysvideo_39 ; bh = 3, pixel r/w
10101                              <1>     ;ja     short sysvideo_17
10102                              <1>     ; 23/07/2022
10103 0000F5A1 7407                <1>     je      short sysvideo_16_0
10104 0000F5A3 7721                <1>     ja      short sysvideo_17
10105 0000F5A5 E94EFCFFFF          <1>     jmp     sysvideo_39
10106                              <1> sysvideo_16_0:
10107                              <1>     ; BH = 4
10108                              <1>     ; Direct User Access for CGA video memory.
10109                              <1>     ; Setup user's page tables for direct access to 0B8000h.
10110                              <1>     ;
10111                              <1>     ; Permission checks are not implemented yet !
10112                              <1>     ; (11/07/2016)
10113                              <1>
10114 0000F5AA B800800B00          <1>     mov     eax, 0B8000h
10115                              <1>     ;mov    ecx, 8 ; 8 pages (8*4K=32K)
10116                              <1>     ; 11/08/2022
10117 0000F5AF 31C9                <1>     xor     ecx, ecx
10118 0000F5B1 B108                <1>     mov     cl, 8
10119 0000F5B3 89C3                <1>     mov     ebx, eax ; 12/05/2017 ; virtual = physical
10120 0000F5B5 E8096AFFFF          <1>     call    direct_memory_access
10121                              <1>     ;jc     sysret
10122 0000F5BA 7205                <1>     jc      short sysvideo_39_79 ; 06/03/2021
10123                              <1>     ; eax = 0B8000h if there is not an error
10124 0000F5BC A3[348E0100]        <1>     mov     [u.r0], eax
10125                              <1> sysvideo_39_79: ; 08/01/2021
10126 0000F5C1 E9B8D4FFFF          <1>     jmp     sysret
10127                              <1>
10128                              <1> sysvideo_17:
10129                              <1>     ; 23/07/2022
10130                              <1>     ; 23/12/2020
```

```
10131                                   <1>     ; 11/12/2020
10132                                   <1>     ; 10/12/2020
10133                                   <1>     ; 23/11/2020
10134 0000F5C6 80FF06                  <1>     cmp     bh, 6
10135 0000F5C9 7424                    <1>     je      short sysvideo_17_0 ; 23/07/2022
10136 0000F5CB 7205                    <1>     jb      short sysvideo_18 ; 23/07/2022
10137 0000F5CD E95E010000              <1>     jmp     sysvideo_20 ; ja
10138                                   <1>
10139                                   <1>     ; 23/07/2022
10140                                   <1> sysvideo_18:
10141                                   <1>     ; BH = 5
10142                                   <1>     ; Direct User Access for VGA video memory.
10143                                   <1>     ; Setup user's page tables for direct access to 0A0000h.
10144                                   <1>     ;
10145                                   <1>     ; Permission checks are not implemented yet !
10146                                   <1>     ; (11/07/2016)
10147                                   <1>
10148 0000F5D2 B800000A00              <1>     mov     eax, 0A0000h
10149 0000F5D7 B910000000              <1>     mov     ecx, 16 ; 16 pages (16*4K=64K)
10150 0000F5DC 89C3                    <1>     mov     ebx, eax ; 12/05/2017 ; virtual = physical
10151 0000F5DE E8E069FFFF              <1>     call    direct_memory_access
10152                                   <1>     ;jc     sysret
10153                                   <1>     ; 23/07/2022
10154 0000F5E3 7205                    <1>     jc      short sysvideo_18_0
10155                                   <1>     ; eax = 0A0000h if there is not an error
10156 0000F5E5 A3[348E0100]            <1>     mov     [u.r0], eax
10157                                   <1> sysvideo_18_0:
10158 0000F5EA E98FD4FFFF              <1>     jmp     sysret
10159                                   <1>
10160                                   <1> sysvideo_17_0:
10161                                   <1>     ; BH = 6
10162                                   <1>     ; Direct User Access to Linear Frame Buffer.
10163                                   <1>     ; Setup user's page tables for direct access to LFB.
10164                                   <1>     ;
10165                                   <1>     ; Permission checks are not implemented yet !
10166                                   <1>     ; (10/12/2020)
10167                                   <1>
10168 0000F5EF 80FBFF                  <1>     cmp     bl, 0FFh ; current video mode
10169 0000F5F2 722C                    <1>     jb      short sysvideo_17_2 ; for desired video mode
10170                                   <1>
10171 0000F5F4 381D[1E680000]          <1>     cmp     [CRT_MODE], bl ; VESA VBE video mode ?
10172 0000F5FA 750E                    <1>     jne     short sysvideo_17_1
10173 0000F5FC 668B0D[EE9C0100]        <1>     mov     cx, [video_mode]
10174 0000F603 6681E1FF01              <1>     and     cx, 1FFh
10175 0000F608 EB29                    <1>     jmp     short sysvideo_17_3
10176                                   <1> sysvideo_17_1:
10177                                   <1>     ; 11/12/2020
10178 0000F60A 88DF                    <1>     mov     bh, bl ; 0FFh
10179 0000F60C 8A1D[1E680000]          <1>     mov     bl, [CRT_MODE] ; VGA/CGA video mode
10180 0000F612 8B2D[308E0100]          <1>     mov     ebp, [u.usp]  ; ebp points to user's registers
10181                                   <1>     ; 23/12/2020
10182 0000F618 895D10                  <1>     mov     [ebp+16], ebx ; return to user with EBX value
10183 0000F61B E95ED4FFFF              <1>     jmp     sysret ; return to user with EAX = 0
10184                                   <1> sysvideo_17_2:
10185                                   <1>     ; bl = VESA video mode - 100h
10186 0000F620 B701                    <1>     mov     bh, 1 ; bx = 1XXh
10187 0000F622 53                      <1>     push    ebx ; requested vesa video mode
10188 0000F623 E88B44FFFF              <1>     call    vbe_biosfn_return_current_mode
10189 0000F628 59                      <1>     pop     ecx ; requested vesa video mode
10190 0000F629 6681E3FF01              <1>     and     bx, 1FFh
10191 0000F62E 6639D9                  <1>     cmp     cx, bx
10192 0000F631 7564                    <1>     jne     short sysvideo_17_8
10193                                   <1> sysvideo_17_3:
10194 0000F633 663B0D[FA9C0100]        <1>     cmp     cx, [LFB_Info+LFBINFO.mode]
10195 0000F63A 755B                    <1>     jne     short sysvideo_17_8
10196                                   <1> sysvideo_17_4:
10197                                   <1>     ; 11/12/2020
10198 0000F63C A1[FC9C0100]            <1>     mov     eax, [LFB_Info+LFBINFO.LFB_addr]
10199                                   <1>     ; 21/12/2020
10200 0000F641 09C0                    <1>     or      eax, eax
10201 0000F643 744D                    <1>     jz      short sysvideo_17_7
10202                                   <1>     ;
10203 0000F645 8B0D[009D0100]          <1>     mov     ecx, [LFB_Info+LFBINFO.LFB_size] ; buff size
10204 0000F64B 89C3                    <1>     mov     ebx, eax ; user's address = physical address
10205                                   <1>     ;push   ebx
10206 0000F64D 51                      <1>     push    ecx
10207                                   <1>     ; 21/12/2020
10208 0000F64E 81C1FF0F0000            <1>     add     ecx, 4095  ; PAGESIZE - 1
10209                                   <1>     ; 14/12/2020
10210 0000F654 C1E90C                  <1>     shr     ecx, 12  ; convert bytes to pages
10211 0000F657 E86769FFFF              <1>     call    direct_memory_access
10212 0000F65C 5A                      <1>     pop     edx  ; linear frame buffer size in bytes
10213                                   <1>     ;pop    eax  ; linear frame buffer address (physical)
10214 0000F65D 7233                    <1>     jc      short sysvideo_17_7 ; [u.r0] = eax = 0
10215                                   <1> sysvideo_17_5:
10216 0000F65F 668B0D[069D0100]        <1>     mov     cx, [LFB_Info+LFBINFO.Y_res] ; screen height
10217 0000F666 C1E110                  <1>     shl     ecx, 16
10218 0000F669 668B0D[049D0100]        <1>     mov     cx,  [LFB_Info+LFBINFO.X_res] ; screen width
10219 0000F670 31DB                    <1>     xor     ebx, ebx
10220 0000F672 8A1D[089D0100]          <1>     mov     bl, [LFB_Info+LFBINFO.bpp] ; bits per pixel
10221 0000F678 8A3D[FA9C0100]          <1>     mov     bh, [LFB_Info+LFBINFO.mode] ; XX part of 1XXh
10222                                   <1> sysvideo_26_4: ; 23/12/2020
10223 0000F67E 8B2D[308E0100]          <1>     mov     ebp, [u.usp]  ; ebp points to user's registers
10224 0000F684 895514                  <1>     mov     [ebp+20], edx ; return to user with EDX value
10225 0000F687 895D10                  <1>     mov     [ebp+16], ebx ; EBX
10226 0000F68A 894D18                  <1>     mov     [ebp+24], ecx ; ECX
10227                                   <1> sysvideo_17_6:
10228 0000F68D A3[348E0100]            <1>     mov     [u.r0], eax ; LFB address
10229                                   <1> sysvideo_17_7:
10230 0000F692 E9E7D3FFFF              <1>     jmp     sysret
10231                                   <1> sysvideo_17_8:
10232                                   <1>     ; cx = mode
10233                                   <1>     ; 21/12/2020
10234 0000F697 80CD40                  <1>     or      ch, 40h  ; Linear frame buffer flag
10235 0000F69A E83742FFFF              <1>     call    _vbe_biosfn_return_mode_info
10236 0000F69F 72F1                    <1>     jc      short sysvideo_17_7
10237 0000F6A1 EB99                    <1>     jmp     short sysvideo_17_4
10238                                   <1>
10239                                   <1> sysvideo_19:
10240                                   <1>     ; 23/07/2022
10241                                   <1>     ; 22/01/2021
10242                                   <1>     ; 12/12/2020
10243                                   <1>     ; 11/12/2020
10244                                   <1>     ; 23/11/2020
10245                                   <1>     ; BH = 7
10246                                   <1>     ; Get (Super/Extended VGA) mode
10247                                   <1>     ; and Linear Frame Buffer info.
10248                                   <1>
10249                                   <1>     ; 22/01/2021
10250 0000F6A3 B3FF                    <1>     mov     bl, 0FFh
10251                                   <1>     ; 11/12/2020
10252                                   <1>     ;cmp    byte [CRT_MODE], 0FFh ; (extended mode?)
10253                                   <1>     ; 22/01/2021
10254 0000F6A5 381D[1E680000]          <1>     cmp     [CRT_MODE], bl  ; 0FFh
```

```
10255                                   <1>     ;jb    short sysvideo_17_1 ; not a VESA VBE mode
10256                                   <1>     ; 23/07/2022
10257                                   <1>     ; 12/12/2020
10258 0000F6AB 7305                     <1>     jnb    short sysvideo_19_0
10259                                   <1>     ; 23/07/2022
10260 0000F6AD E958FFFFFF               <1>     jmp    sysvideo_17_1
10261                                   <1>
10262                                   <1> sysvideo_19_0:
10263 0000F6B2 E8FC43FFFF               <1>     call   vbe_biosfn_return_current_mode
10264 0000F6B7 6681E3FF01               <1>     and    bx, 1FFh
10265 0000F6BC 663B1D[FA9C0100]         <1>     cmp    bx, [LFB_Info+LFBINFO.mode]
10266 0000F6C3 750D                     <1>     jne    short sysvideo_19_2
10267                                   <1> sysvideo_19_1:
10268 0000F6C5 A1[FC9C0100]             <1>     mov    eax, [LFB_Info+LFBINFO.LFB_addr]
10269 0000F6CA 8B15[009D0100]           <1>     mov    edx, [LFB_Info+LFBINFO.LFB_size]
10270 0000F6D0 EB8D                     <1>     jmp    sysvideo_17_5
10271                                   <1> sysvideo_19_2:
10272 0000F6D2 E8FF41FFFF               <1>     call   _vbe_biosfn_return_mode_info
10273 0000F6D7 73EC                     <1>     jnc    short sysvideo_19_1
10274 0000F6D9 E9A0D3FFFF               <1>     jmp    sysret
10275                                   <1>
10276                                   <1> sysvideo_20_1:
10277                                   <1>     ; cx = vesa video mode
10278 0000F6DE 6689C8                   <1>     mov    ax, cx
10279 0000F6E1 663D0001                 <1>     cmp    ax, 100h
10280 0000F6E5 725C                     <1>     jb     short sysvideo_20_0 ; VGA/CGA mode
10281 0000F6E7 663DFF01                 <1>     cmp    ax, 1FFh
10282                                   <1>     ;ja    short sysvideo_20_4 ; not valid
10283 0000F6EB 773E                     <1>     ja     short sysvideo_20_3
10284 0000F6ED 50                       <1>     push   eax
10285 0000F6EE 6689C3                   <1>     mov    bx, ax
10286 0000F6F1 66B8024F                 <1>     mov    ax, 4F02h
10287                                   <1>
10288                                   <1>     ; simulate _int10h (int 31h) for func 4F02h
10289                                   <1>     ;pushfd
10290                                   <1>     ;push  cs
10291                                   <1>     ;push  sysvideo_20_1_retn
10292                                   <1>     ;push  es ; *
10293                                   <1>     ;push  ds ; ** ; SAVE WORK AND PARAMETER REGISTERS
10294                                   <1>     ;jmp   VBE_func
10295                                   <1> ;sysvideo_20_1_retn:
10296                                   <1>
10297 0000F6F5 E82A20FFFF               <1>     call   _int10h ; simulate int 10h (int 31h)
10298                                   <1>
10299 0000F6FA 6683F84F                 <1>     cmp    ax, 004Fh
10300 0000F6FE 58                       <1>     pop    eax
10301 0000F6FF 752A                     <1>     jne    short sysvideo_20_3 ; error
10302                                   <1>     ;pop   eax
10303 0000F701 40                       <1>     inc    eax
10304 0000F702 A3[348E0100]             <1>     mov    [u.r0], eax ; video mode + 1
10305 0000F707 09D2                     <1>     or     edx, edx ; is LFBINFO requested by user ?
10306                                   <1>     ;jz    short sysvideo_20_4
10307 0000F709 7420                     <1>     jz     short sysvideo_20_3 ; no
10308                                   <1>
10309                                   <1>     ; 11/12/2020
10310                                   <1>     ; Check LFBINFO table/structure
10311                                   <1>     ; (it is set by vbe2 'vbe_biosfn_set_mode'
10312                                   <1>     ; but if vbe3 vbios pmi is in use,
10313                                   <1>     ; it will not set LFBINFO table)
10314                                   <1>
10315 0000F70B 52                       <1>     push   edx
10316 0000F70C 48                       <1>     dec    eax  ; video mode
10317 0000F70D BE[FA9C0100]             <1>     mov    esi, LFB_Info
10318 0000F712 663B06                   <1>     cmp    ax, [esi+LFBINFO.mode]
10319 0000F715 7407                     <1>     je     short sysvideo_20_2
10320                                   <1>
10321 0000F717 E8BA41FFFF               <1>     call   _vbe_biosfn_return_mode_info
10322                                   <1>     ;jnc   short sysvideo_20_2
10323 0000F71C 723B                     <1>     jc     short sysvideo_20_4 ; edx = 0
10324                                   <1>
10325                                   <1>     ;; clear LFBINFO table for invalidating
10326                                   <1>     ;mov   ecx, LFBINFO.size ; 16
10327                                   <1>     ;mov   edi, esi ; LFB_Info table address
10328                                   <1>     ;xor   eax, eax
10329                                   <1>     ;rep   stosb
10330                                   <1>
10331                                   <1> sysvideo_20_2:
10332                                   <1>     ;pop   ecx
10333                                   <1>     ;mov   edi, ecx ; user buffer
10334 0000F71E 5F                       <1>     pop    edi
10335 0000F71F B910000000               <1>     mov    ecx, LFBINFO.size ; 16
10336 0000F724 E8C1130000               <1>     call   transfer_to_user_buffer ; fast transfer
10337 0000F729 722F                     <1>     jc     short sysvideo_20_5
10338                                   <1>
10339                                   <1>     ;jmp   sysret
10340                                   <1> sysvideo_20_3:
10341                                   <1>     ;pop   eax ; [u.r0] = 0
10342                                   <1> ;sysvideo_20_4:
10343 0000F72B E94ED3FFFF               <1>     jmp    sysret
10344                                   <1>
10345                                   <1>     ; 23/07/2022
10346                                   <1> sysvideo_20:
10347                                   <1>     ; 11/12/2020
10348                                   <1>     ; 23/11/2020
10349 0000F730 80FF08                   <1>     cmp    bh, 8
10350                                   <1>     ; 08/08/2022
10351                                   <1>     ;jb    short sysvideo_19 ; video mode & lfb info
10352 0000F733 7407                     <1>     je     short sysvideo_20_6
10353                                   <1>     ; 23/07/2022
10354 0000F735 7733                     <1>     ja     short sysvideo_21 ; 12/12/2020
10355                                   <1>     ; 08/08/2022 - jb
10356 0000F737 E967FFFFFF               <1>     jmp    sysvideo_19
10357                                   <1> sysvideo_20_6:
10358                                   <1>     ; BH = 8
10359                                   <1>     ; Set (Super/Extended VGA) mode & return LFB info
10360                                   <1>
10361                                   <1>     ; 11/12/2020
10362 0000F73C 80FBFF                   <1>     cmp    bl, 0FFh  ; CGA/VGA mode ?
10363 0000F73F 739D                     <1>     jnb    short sysvideo_20_1
10364                                   <1>
10365                                   <1>     ;xor   ah, ah
10366 0000F741 88D8                     <1>     mov    al, bl
10367                                   <1> sysvideo_20_0:
10368 0000F743 E8DC1FFFFF               <1>     call   _int10h  ; uses vbe3 pmi32 option
10369 0000F748 83F8FF                   <1>     cmp    eax, 0FFFFFFFFh ; -1
10370 0000F74B 74DE                     <1>     je     short sysvideo_20_3 ; error
10371                                   <1>
10372                                   <1>     ; 11/12/2020
10373                                   <1>     ; alternative (it does not use vbe3 pmi32)
10374                                   <1>     ;push  eax
10375                                   <1>     ;call  _set_mode
10376                                   <1>     ;pop   eax
10377                                   <1>     ;jc    short sysvideo_20_3
10378                                   <1>
```

```
10379                                   <1>     ;inc    eax
10380 0000F74D FEC0                     <1>     inc     al
10381                                   <1>     ;mov    [u.r0], ax ; video mode + 1
10382 0000F74F A2[348E0100]             <1>     mov     [u.r0], al
10383 0000F754 E925D3FFFF               <1>     jmp     sysret
10384                                   <1>
10385                                   <1> sysvideo_20_4:
10386 0000F759 5A                       <1>     pop     edx
10387                                   <1> sysvideo_20_5:
10388 0000F75A 31D2                     <1>     xor     edx, edx ; 0
10389                                   <1>     ; edx = 0 -> invalid LFBINFO data
10390 0000F75C 8B2D[308E0100]           <1>     mov     ebp, [u.usp]  ; ebp points to user's registers
10391 0000F762 895514                   <1>     mov     [ebp+20], edx ; return to user with EDX value
10392 0000F765 E914D3FFFF               <1>     jmp     sysret
10393                                   <1>
10394                                   <1> sysvideo_21:
10395                                   <1>     ; 23/07/2022
10396                                   <1>     ; 04/01/2021
10397                                   <1>     ; 03/12/2020
10398 0000F76A 80FF0A                   <1>     cmp     bh, 10
10399                                   <1>     ;jb     sysvideo_22 ; VESA VBE3 pmi parms
10400                                   <1>     ; 23/07/2022
10401 0000F76D 723F                     <1>     jb      short sysvideo_21_19
10402                                   <1>     ; 23/12/2020
10403                                   <1>     ;je     sysvideo_26 ; Video memory mapping
10404                                   <1>     ; 23/07/2022
10405 0000F76F 7442                     <1>     je      short sysvideo_21_20
10406                                   <1>
10407                                   <1>     ; 04/01/2020
10408 0000F771 80FF0B                   <1>     cmp     bh, 11
10409                                   <1>     ;ja     sysvideo_27
10410                                   <1>     ; 23/07/2022
10411 0000F774 7742                     <1>     ja      short sysvideo_21_21
10412                                   <1>
10413                                   <1>     ; BH = 11
10414                                   <1>     ; set/read DAC color registers (for 8bpp)
10415                                   <1>
10416 0000F776 80FB04                   <1>     cmp     bl, 4
10417                                   <1>     ;jnb    sysvideo_21_7  ; BMP file type palette
10418                                   <1>                            ; handling
10419                                   <1>     ; 23/07/2022
10420                                   <1>     ;jnb    short sysvideo_21_7
10421                                   <1>     ; 08/08/2022
10422 0000F779 7205                     <1>     jb      short sysvideo_21_18
10423 0000F77B E989000000               <1>     jmp     sysvideo_21_7
10424                                   <1> sysvideo_21_18:
10425 0000F780 F6C301                   <1>     test    bl, 1
10426 0000F783 7563                     <1>     jnz     short sysvideo_21_4 ; set/write DAC colors
10427                                   <1>
10428                                   <1>     ; Read DAC color register or all DAC color registers
10429 0000F785 F6C302                   <1>     test    bl, 2  ; read single DAC color register
10430 0000F788 7433                     <1>     jz      short sysvideo_21_2 ; read all DAC color regs
10431                                   <1>
10432                                   <1>     ; read single DAC color register
10433                                   <1>     ; CL = DAC color register (index)
10434                                   <1>
10435 0000F78A 66BAC703                 <1>     mov     dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
10436 0000F78E 88C8                     <1>     mov     al, cl  ; DAC color register
10437 0000F790 31C9                     <1>     xor     ecx, ecx ; (this may not be necessary)
10438 0000F792 EE                       <1>     out     dx, al
10439                                   <1>     ;mov    dx, 3C9h ; VGAREG_DAC_DATA
10440 0000F793 B2C9                     <1>     mov     dl, 0C9h
10441 0000F795 EC                       <1>     in      al, dx
10442 0000F796 88C4                     <1>     mov     ah, al  ; red
10443 0000F798 EC                       <1>     in      al, dx
10444 0000F799 88C1                     <1>     mov     cl, al  ; green
10445 0000F79B EC                       <1>     in      al, dx
10446 0000F79C 88C5                     <1>     mov     ch, al  ; blue
10447 0000F79E C1E108                   <1>     shl     ecx, 8
10448 0000F7A1 88E1                     <1>     mov     cl, ah  ; red
10449                                   <1>     ; CL = Red, CH = Green, byte 3 = Blue, byte 4 = 0
10450                                   <1> sysvideo_21_0:
10451 0000F7A3 890D[348E0100]           <1>     mov     [u.r0], ecx
10452                                   <1> sysvideo_21_1:
10453 0000F7A9 E9D0D2FFFF               <1>     jmp     sysret
10454                                   <1> sysvideo_21_19:
10455                                   <1>     ; 23/07/2022
10456 0000F7AE E97A010000               <1>     jmp     sysvideo_22
10457                                   <1> sysvideo_21_20:
10458                                   <1>     ; 23/07/2022
10459 0000F7B3 E930020000               <1>     jmp     sysvideo_26
10460                                   <1> sysvideo_21_21:
10461                                   <1>     ; 23/07/2022
10462 0000F7B8 E9B6020000               <1>     jmp     sysvideo_27
10463                                   <1> sysvideo_21_2:
10464                                   <1>     ; read all DAC color registers
10465 0000F7BD 89CB                     <1>     mov     ebx, ecx ; user's buffer address
10466 0000F7BF BF00600900               <1>     mov     edi, VBE3STACKADDR
10467 0000F7C4 89FE                     <1>     mov     esi, edi
10468                                   <1>     ;mov    ecx, 768 ; 256*3
10469                                   <1>     ; 08/08/2022
10470 0000F7C6 29C9                     <1>     sub     ecx, ecx
10471 0000F7C8 B503                     <1>     mov     ch, 3
10472                                   <1>     ; ecx = 768 = 300h
10473                                   <1>     ;push   ecx
10474 0000F7CA 66BAC703                 <1>     mov     dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
10475 0000F7CE 28C0                     <1>     sub     al, al ; 0
10476 0000F7D0 EE                       <1>     out     dx, al
10477                                   <1>     ;mov    dx, 3C9h ; VGAREG_DAC_DATA
10478 0000F7D1 B2C9                     <1>     mov     dl, 0C9h
10479                                   <1> sysvideo_21_3:
10480 0000F7D3 EC                       <1>     in      al, dx
10481 0000F7D4 AA                       <1>     stosb
10482 0000F7D5 EC                       <1>     in      al, dx
10483 0000F7D6 AA                       <1>     stosb
10484 0000F7D7 EC                       <1>     in      al, dx
10485 0000F7D8 AA                       <1>     stosb
10486 0000F7D9 E2F8                     <1>     loop    sysvideo_21_3
10487                                   <1>     ;pop    ecx
10488                                   <1>     ; 18/08/2022
10489 0000F7DB B503                     <1>     mov     ch, 3
10490                                   <1>     ; ecx = 768 = 300h
10491                                   <1>
10492 0000F7DD 89DF                     <1>     mov     edi, ebx ; user's buffer address
10493                                   <1>     ;mov    esi, VBE3STACKADDR
10494                                   <1>     ;mov    ecx, 256*3 = 768
10495 0000F7DF E806130000               <1>     call    transfer_to_user_buffer
10496 0000F7E4 72C3                     <1>     jc      short sysvideo_21_1
10497                                   <1>     ;mov    [u.r0], ecx ; actual transfer count
10498 0000F7E6 EBBB                     <1>     jmp     short sysvideo_21_0
10499                                   <1>
10500                                   <1> sysvideo_21_4:
10501                                   <1>     ; Set/Write DAC color register or all registers
10502 0000F7E8 F6C302                   <1>     test    bl, 2 ; write/set single DAC color register
```

```
10503 0000F7EB 7456              <1>     jz      short sysvideo_21_5 ; set all DAC color regs
10504                            <1>
10505                            <1>     ; set single DAC color register
10506                            <1>     ; CL = DAC color register (index)
10507                            <1>     ; (byte 1 = Red, byte 2 = Green, byte 3 = Blue)
10508                            <1>
10509 0000F7ED 66BAC803         <1>     mov     dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
10510 0000F7F1 89C8              <1>     mov     eax, ecx ; DAC color register (index)
10511 0000F7F3 C1E910            <1>     shr     ecx, 16  ; CL = green, AH = Red
10512 0000F7F6 EE                <1>     out     dx, al
10513                            <1>     ;mov    dx, 3C9h ; VGAREG_DAC_DATA
10514 0000F7F7 FEC2              <1>     inc     dl
10515 0000F7F9 88E0              <1>     mov     al, ah  ; Red
10516 0000F7FB EE                <1>     out     dx, al
10517 0000F7FC 88C8              <1>     mov     al, cl  ; Green
10518 0000F7FE EE                <1>     out     dx, al
10519 0000F7FF 88E8              <1>     mov     al, ch  ; Blue
10520 0000F801 EE                <1>     out     dx, al
10521                            <1>     ;rol    ecx, 8
10522 0000F802 C1E108            <1>     shl     ecx, 8  ; 21/02/2021
10523 0000F805 88E1              <1>     mov     cl, ah  ; Red
10524                            <1>             ; ecx = 00BBGGRRh
10525 0000F807 EB9A              <1>     jmp     short sysvideo_21_0
10526                            <1>
10527                            <1>     ; 23/07/2022
10528                            <1> sysvideo_21_7:
10529                            <1>     ; BMP file type palette handling
10530                            <1>
10531 0000F809 F6C301            <1>     test    bl, 1
10532                            <1>     ;jnz    short sysvideo_21_12 ; set/write DAC colors
10533                            <1>     ; 08/08/2022
10534 0000F80C 7405              <1>     jz      short sysvideo_21_16
10535 0000F80E E9A3000000        <1>     jmp     sysvideo_21_12
10536                            <1>
10537                            <1> sysvideo_21_16:
10538                            <1>     ; Read DAC color register or all DAC color registers
10539 0000F813 F6C302            <1>     test    bl, 2   ; read single DAC color register
10540 0000F816 7460              <1>     jz      short sysvideo_21_10 ; read all DAC color regs
10541                            <1>
10542                            <1>     ; read single DAC color register
10543                            <1>     ; CL = DAC color register (index)
10544                            <1>
10545 0000F818 66BAC703          <1>     mov     dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
10546 0000F81C 88C8              <1>     mov     al, cl  ; DAC color register
10547 0000F81E 31C9              <1>     xor     ecx, ecx
10548 0000F820 EE                <1>     out     dx, al
10549                            <1>     ;mov    dx, 3C9h ; VGAREG_DAC_DATA
10550 0000F821 B2C9              <1>     mov     dl, 0C9h
10551 0000F823 EC                <1>     in      al, dx
10552 0000F824 C0E002            <1>     shl     al, 2
10553 0000F827 88C5              <1>     mov     ch, al  ; red
10554 0000F829 EC                <1>     in      al, dx
10555 0000F82A C0E002            <1>     shl     al, 2
10556 0000F82D 88C1              <1>     mov     cl, al  ; green
10557 0000F82F EC                <1>     in      al, dx
10558 0000F830 C0E002            <1>     shl     al, 2
10559                            <1>     ; 21/02/2021
10560 0000F833 C1E108            <1>     shl     ecx, 8
10561 0000F836 88C1              <1>     mov     cl, al  ; blue
10562                            <1>     ; CL = Blue, CH = Green, byte 3 = Red, byte 4 = 0
10563                            <1> sysvideo_21_8:
10564 0000F838 890D[348E0100]    <1>     mov     [u.r0], ecx
10565                            <1> sysvideo_21_9:
10566 0000F83E E93BD2FFFF        <1>     jmp     sysret
10567                            <1>
10568                            <1> sysvideo_21_5:
10569                            <1>     ; write/set all DAC color registers
10570 0000F843 89CE              <1>     mov     esi, ecx ; user's buffer address
10571 0000F845 BF00600900        <1>     mov     edi, VBE3STACKADDR
10572 0000F84A 89FB              <1>     mov     ebx, edi
10573 0000F84C B900030000        <1>     mov     ecx, 768 ; 256*3
10574 0000F851 E8DE120000        <1>     call    transfer_from_user_buffer
10575                            <1>     ;jc     short sysvideo_21_1
10576                            <1>     ; 08/08/2022
10577 0000F856 7305              <1>     jnc     short sysvideo_21_17
10578 0000F858 E94CFFFFFF        <1>     jmp     sysvideo_21_1
10579                            <1> sysvideo_21_17:
10580 0000F85D 890D[348E0100]    <1>     mov     [u.r0], ecx ; actual transfer count
10581                            <1>
10582 0000F863 89DE              <1>     mov     esi, ebx ; VBE3STACKADDR
10583 0000F865 66BAC803          <1>     mov     dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
10584 0000F869 28C0              <1>     sub     al, al ; 0
10585 0000F86B EE                <1>     out     dx, al
10586                            <1>     ;mov    dx, 3C9h ; VGAREG_DAC_DATA
10587 0000F86C FEC2              <1>     inc     dl
10588                            <1> sysvideo_21_6:
10589 0000F86E AC                <1>     lodsb
10590 0000F86F EE                <1>     out     dx, al
10591 0000F870 AC                <1>     lodsb
10592 0000F871 EE                <1>     out     dx, al
10593 0000F872 AC                <1>     lodsb
10594 0000F873 EE                <1>     out     dx, al
10595 0000F874 E2F8              <1>     loop    sysvideo_21_6
10596 0000F876 EBC6              <1>     jmp     short sysvideo_21_9
10597                            <1>
10598                            <1> sysvideo_21_10:
10599                            <1>     ; read all DAC color registers
10600 0000F878 89CD              <1>     mov     ebp, ecx ; user's buffer address
10601 0000F87A BF00600900        <1>     mov     edi, VBE3STACKADDR
10602 0000F87F 89FE              <1>     mov     esi, edi
10603 0000F881 B900040000        <1>     mov     ecx, 1024 ; 256*4
10604 0000F886 51                <1>     push    ecx
10605 0000F887 66BAC703          <1>     mov     dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
10606 0000F88B 28C0              <1>     sub     al, al ; 0
10607 0000F88D EE                <1>     out     dx, al
10608                            <1>     ;mov    dx, 3C9h ; VGAREG_DAC_DATA
10609 0000F88E B2C9              <1>     mov     dl, 0C9h
10610                            <1> sysvideo_21_11:
10611 0000F890 31DB              <1>     xor     ebx, ebx
10612 0000F892 EC                <1>     in      al, dx  ; Red
10613 0000F893 C0E002            <1>     shl     al, 2
10614 0000F896 88C7              <1>     mov     bh, al
10615 0000F898 EC                <1>     in      al, dx  ; Green
10616 0000F899 C0E002            <1>     shl     al, 2
10617 0000F89C 88C3              <1>     mov     bl, al
10618 0000F89E EC                <1>     in      al, dx  ; Blue
10619 0000F89F C0E002            <1>     shl     al, 2
10620 0000F8A2 C1E308            <1>     shl     ebx, 8
10621 0000F8A5 89D8              <1>     mov     eax, ebx ; 00RRGGBBh
10622 0000F8A7 AB                <1>     stosd
10623 0000F8A8 E2E6              <1>     loop    sysvideo_21_11
10624 0000F8AA 59                <1>     pop     ecx
10625                            <1>
10626 0000F8AB 89EF              <1>     mov     edi, ebp ; user's buffer address
```

```
10627                                   <1>     ;mov    esi, VBE3STACKADDR
10628                                   <1>     ;mov    ecx, 1024 = 4*256
10629 0000F8AD E838120000               <1>     call    transfer_to_user_buffer
10630 0000F8B2 728A                     <1>     jc      short sysvideo_21_9
10631                                   <1>     ;mov    [u.r0], ecx ; actual transfer count
10632 0000F8B4 EB82                     <1>     jmp     short sysvideo_21_8
10633                                   <1>
10634                                   <1> sysvideo_21_12:
10635                                   <1>     ; Set/Write DAC color register or all registers
10636 0000F8B6 F6C302                   <1>     test    bl, 2 ; write/set single DAC color register
10637 0000F8B9 742A                     <1>     jz      short sysvideo_21_13 ; set all DAC color regs
10638                                   <1>
10639                                   <1>     ; set single DAC color register
10640                                   <1>     ; CL = DAC color register (index)
10641                                   <1>     ; (byte 1 = Blue, byte 2 = Green, byte 3 = Red)
10642                                   <1>
10643 0000F8BB 66BAC803                <1>     mov     dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
10644 0000F8BF 88C8                     <1>     mov     al, cl  ; DAC color register (index)
10645 0000F8C1 88EC                     <1>     mov     ah, ch  ; Blue
10646 0000F8C3 C1E910                   <1>     shr     ecx, 16
10647 0000F8C6 EE                       <1>     out     dx, al
10648                                   <1>     ;mov    dx, 3C9h ; VGAREG_DAC_DATA
10649 0000F8C7 FEC2                     <1>     inc     dl
10650 0000F8C9 88E8                     <1>     mov     al, ch  ; Red
10651 0000F8CB C0E802                   <1>     shr     al, 2
10652 0000F8CE EE                       <1>     out     dx, al
10653 0000F8CF 88C8                     <1>     mov     al, cl  ; Green
10654 0000F8D1 C0E802                   <1>     shr     al, 2
10655 0000F8D4 EE                       <1>     out     dx, al
10656 0000F8D5 88E0                     <1>     mov     al, ah  ; Blue
10657 0000F8D7 C0E802                   <1>     shr     al, 2
10658 0000F8DA EE                       <1>     out     dx, al
10659                                   <1>     ;rol    ecx, 8
10660 0000F8DB C1E108                   <1>     shl     ecx, 8 ; 21/02/2021
10661 0000F8DE 88E1                     <1>     mov     cl, ah
10662 0000F8E0 E953FFFFFF               <1>     jmp     sysvideo_21_8 ; 08/08/2022
10663                                   <1>
10664                                   <1> sysvideo_21_13:
10665                                   <1>     ; write/set all DAC color registers
10666 0000F8E5 89CE                     <1>     mov     esi, ecx ; user's buffer address
10667 0000F8E7 BF00600900               <1>     mov     edi, VBE3STACKADDR
10668 0000F8EC 89FB                     <1>     mov     ebx, edi
10669 0000F8EE B900040000               <1>     mov     ecx, 1024 ; 256*4
10670 0000F8F3 E83C120000               <1>     call    transfer_from_user_buffer
10671 0000F8F8 722E                     <1>     jc      short sysvideo_21_15
10672 0000F8FA 890D[348E0100]           <1>     mov     [u.r0], ecx ; actual transfer count
10673                                   <1>
10674 0000F900 89DE                     <1>     mov     esi, ebx ; VBE3STACKADDR
10675 0000F902 66BAC803                 <1>     mov     dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
10676 0000F906 28C0                     <1>     sub     al, al ; 0
10677 0000F908 EE                       <1>     out     dx, al
10678                                   <1>     ;mov    dx, 3C9h ; VGAREG_DAC_DATA
10679 0000F909 FEC2                     <1>     inc     dl
10680                                   <1> sysvideo_21_14:
10681 0000F90B AD                       <1>     lodsd
10682                                   <1>     ; byte 0 = Blue, byte 1 = Green, byte 2 = Red
10683                                   <1>     ; 21/02/2021
10684 0000F90C 89C3                     <1>     mov     ebx, eax ; BL = Blue, BH = Green
10685 0000F90E C1CB08                   <1>     ror     ebx, 8 ; BL = Green, BH = Red
10686 0000F911 88F8                     <1>     mov     al, bh
10687 0000F913 C0E802                   <1>     shr     al, 2
10688 0000F916 EE                       <1>     out     dx, al ; Red
10689 0000F917 88D8                     <1>     mov     al, bl
10690 0000F919 C0E802                   <1>     shr     al, 2
10691 0000F91C EE                       <1>     out     dx, al ; Green
10692 0000F91D C1C308                   <1>     rol     ebx, 8 ; BL = Blue
10693 0000F920 88D8                     <1>     mov     al, bl
10694 0000F922 C0E802                   <1>     shr     al, 2
10695 0000F925 EE                       <1>     out     dx, al ; Blue
10696 0000F926 E2E3                     <1>     loop    sysvideo_21_14
10697                                   <1> sysvideo_21_15:
10698 0000F928 E951D1FFFF               <1>     jmp     sysret
10699                                   <1>
10700                                   <1> sysvideo_22:
10701                                   <1>     ; 28/02/2021
10702                                   <1>     ; 22/01/2021
10703                                   <1>     ; 17/01/2021
10704                                   <1>     ; 04/12/2020
10705                                   <1>     ; 03/12/2020
10706                                   <1>     ; BH = 9
10707                                   <1>     ; Set/Get VESA VBE3 protected mode interface params
10708                                   <1>
10709                                   <1>     ; 22/01/2021
10710                                   <1>     ;cmp    byte [vbe3], 3
10711                                   <1>     ;jne    short sysvideo_25 ; not applicable if
10712                                   <1>                               ; vbe3 compatible video bios
10713                                   <1>                               ; is not detected by kernel
10714 0000F92D 80FB02                   <1>     cmp     bl, 2
10715                                   <1>     ;ja     short sysvideo_25 ; bl > 2 not implemented
10716                                   <1>     ; 17/01/2021
10717 0000F930 7716                     <1>     ja      short sysvideo_22_0 ; srvs flag sub function
10718                                   <1>     ;jb     short sysvideo_23
10719                                   <1>
10720                                   <1>     ; 21/01/2021
10721 0000F932 803D[86090000]03         <1>     cmp     byte [vbe3], 3
10722                                   <1>     ;jne    short sysvideo_25 ; not applicable if
10723                                   <1>                               ; vbe3 compatible video bios
10724                                   <1>                               ; is not detected by kernel
10725 0000F939 75ED                     <1>     jne     short sysvideo_21_15 ; 28/02/2021
10726                                   <1>
10727 0000F93B 80FB01                   <1>     cmp     bl, 1
10728 0000F93E 7673                     <1>     jna     short sysvideo_23
10729                                   <1>
10730 0000F940 8A1D[EC9C0100]           <1>     mov     bl, [pmi32] ; Video bios 32 bit PMI functions
10731 0000F946 EB78                     <1>     jmp     short sysvideo_24
10732                                   <1>
10733                                   <1> sysvideo_22_0:
10734                                   <1>     ; 17/01/2021
10735                                   <1>     ; save/restore video state user permission
10736 0000F948 80FB05                   <1>     cmp     bl, 5
10737 0000F94B 771E                     <1>     ja      short sysvideo_22_2
10738 0000F94D 7208                     <1>     jb      short sysvideo_22_1
10739                                   <1>     ; get srvs flag value/status
10740 0000F94F 8A1D[509D0100]           <1>     mov     bl, [srvsf] ; 0 = disabled, 1 = enabled
10741 0000F955 EB2C                     <1>     jmp     short sysvideo_22_3
10742                                   <1>
10743                                   <1> sysvideo_22_1:
10744                                   <1>     ; permission (root and multi tasking) check
10745 0000F957 E836000000               <1>     call    sysvideo_22_4
10746 0000F95C 736A                     <1>     jnc     short sysvideo_25 ; not permitted !
10747                                   <1>     ; cf = 1
10748 0000F95E 80EB03                   <1>     sub     bl, 3 ; disable = 0, enable = 1
10749                                   <1>     ; 22/01/2021
10750 0000F961 881D[509D0100]           <1>     mov     [srvsf], bl
```

```
10751 0000F967 FEC3                <1>     inc     bl ; 1 = disabled, 2 = enabled
10752 0000F969 EB18                <1>     jmp     short sysvideo_22_3
10753                              <1>
10754                              <1> sysvideo_22_2:
10755 0000F96B 80FB06              <1>     cmp     bl, 6
10756                              <1>     ;ja     short sysvideo_25 ; invalid/unimplemented
10757                              <1>     ; 28/02/2021
10758 0000F96E 7733                <1>     ja      short sysvideo_22_6
10759                              <1>     ; get VESA VBE number/status
10760 0000F970 8A25[86090000]      <1>     mov     ah, [vbe3] ; vbe3 = 3, vbe2 = 2, others = 0
10761 0000F976 A0[87090000]        <1>     mov     al, [vbe2bios] ; bochs/qemu/vbox emulator status
10762 0000F97B 66A3[348E0100]      <1>     mov     [u.r0], ax
10763 0000F981 EB45                <1>     jmp     short sysvideo_25
10764                              <1>
10765                              <1> sysvideo_22_3:
10766                              <1>     ; 22/01/2021
10767 0000F983 8A3D[519D0100]      <1>     mov     bh, [srvso] ; state options (> 80h -> svga)
10768 0000F989 66891D[348E0100]    <1>     mov     [u.r0], bx ; function result is return value
10769 0000F990 EB36                <1>     jmp     short sysvideo_25
10770                              <1>
10771                              <1> sysvideo_22_4:
10772                              <1>     ; 17/01/2021 - permission will be given by root only
10773 0000F992 803D[12830100]00    <1>     cmp     byte [multi_tasking], 0 ; in single user mode
10774 0000F999 7707                <1>     ja      short sysvideo_22_5
10775                              <1>     ; 19/01/2021
10776 0000F99B 803D[868E0100]01    <1>     cmp     byte [u.uid], 1 ; ([u.uid] = 0 -> root)
10777                              <1> sysvideo_22_5:
10778                              <1>     ; [multi_tasking] = 0 & [u.uid] = 0 -> CF = 1
10779                              <1>     ; otherwise -> CF = 0
10780 0000F9A2 C3                  <1>     retn
10781                              <1>
10782                              <1> sysvideo_22_6:
10783                              <1>     ; 28/02/2021
10784 0000F9A3 80FB09              <1>     cmp     bl, 9
10785 0000F9A6 7720                <1>     ja      short sysvideo_25 ; invalid/unimplemented
10786 0000F9A8 7436                <1>     je      short sysvideo_22_9
10787 0000F9AA 80FB08              <1>     cmp     bl, 8
10788 0000F9AD 721E                <1>     jb      short sysvideo_22_7
10789                              <1>
10790                              <1>     ; BL = 8
10791                              <1>     ; Set default true color bpp to 24
10792                              <1>
10793 0000F9AF B318                <1>     mov     bl, 24
10794                              <1>     ;mov    [truecolor], al       ; 24bpp (RRGGBBh)
10795                              <1>     ;mov    [u.r0], al
10796                              <1>     ;jmp    short sysvideo_25
10797 0000F9B1 EB25                <1>     jmp     short sysvideo_22_8
10798                              <1>
10799                              <1> sysvideo_23:
10800                              <1>     ; 17/01/2021
10801                              <1>     ; permission (root and multi tasking) check
10802 0000F9B3 E8DAFFFFFF          <1>     call    sysvideo_22_4
10803 0000F9B8 730E                <1>     jnc     short sysvideo_25 ; not permitted !
10804                              <1>
10805 0000F9BA 881D[EC9C0100]      <1>     mov     [pmi32], bl ; 1 = enabled, 0 = disabled
10806                              <1> sysvideo_24:
10807 0000F9C0 FEC3                <1>     inc     bl
10808                              <1> sysvideo_22_10:    ; 28/02/2021
10809 0000F9C2 881D[348E0100]      <1>     mov     [u.r0], bl ; function result is return value
10810                              <1> sysvideo_25:
10811 0000F9C8 E9B1D0FFFF          <1>     jmp     sysret
10812                              <1>
10813                              <1> sysvideo_22_7:
10814                              <1>     ; BL = 7
10815                              <1>     ; Set default true color bpp to 32
10816                              <1>     ; (it will set if [VBE3]=3)
10817                              <1>
10818                              <1>     ; Note: This sub function is used to set 24bpp
10819                              <1>     ; VESA VBE video modes to 32bpp.. because,
10820                              <1>     ; old hardware uses 24 bpp but new video hardware
10821                              <1>     ; uses 32bpp for same VESA VBE truecolor modes.
10822                              <1>     ; (For example: VBE mode 112h is 640*480, 24bpp but
10823                              <1>     ; new hardware uses/apply it as 640*480, 32bpp.)
10824                              <1>     ; So, TRDOS 386 v2.0.3 kernel will check [truecolor]
10825                              <1>     ; status is 32 bpp or not and it will change 24bpp
10826                              <1>     ; to 32bpp if default [truecolor] value is 32, for
10827                              <1>     ; same video mode number.
10828                              <1>
10829 0000F9CD 803D[86090000]03    <1>     cmp     byte [vbe3], 3
10830 0000F9D4 75F2                <1>     jne     short sysvideo_25 ; Only applicable
10831                              <1>                               ; for VBE3 video hardware!
10832 0000F9D6 B320                <1>     mov     bl, 32
10833                              <1> sysvideo_22_8:
10834 0000F9D8 881D[43730100]      <1>     mov     [truecolor], bl       ; 32bpp (00RRGGBBh)
10835                              <1>     ;mov    [u.r0], bl
10836                              <1>     ;jmp    short sysvideo_25
10837 0000F9DE EBE2                <1>     jmp     short sysvideo_22_10
10838                              <1>
10839                              <1> sysvideo_22_9:
10840                              <1>     ; BL = 9
10841                              <1>     ; Return default true color bpp
10842 0000F9E0 8A1D[43730100]      <1>     mov     bl, [truecolor]
10843 0000F9E6 EBDA                <1>     jmp     short sysvideo_22_10
10844                              <1> ;sysvideo_22_10:
10845                              <1>     ;mov    [u.r0], bl
10846                              <1>     ;jmp    sysret
10847                              <1>
10848                              <1> sysvideo_26:
10849                              <1>     ; 23/12/2020
10850                              <1>     ; BH = 10
10851                              <1>     ; Map video memory to user's buffer
10852                              <1>     ; (multiuser/owner r/w permisions are ignored
10853                              <1>     ;  for current TRDOS 386 version !)
10854                              <1>
10855 0000F9E8 6681E100F0          <1>     and     cx, ~4095  ; clear low 12 bits
10856 0000F9ED 09C9                <1>     or      ecx, ecx ; start address of user's buffer
10857 0000F9EF 74D7                <1>     jz      short sysvideo_25 ; error !
10858                              <1>
10859 0000F9F1 80FB01              <1>     cmp     bl, 1  ; VGA memory mapping ?
10860 0000F9F4 740E                <1>     je      short sysvideo_26_1
10861 0000F9F6 7718                <1>     ja      short sysvideo_26_2
10862                              <1> sysvideo_26_0:
10863                              <1>     ; BL = 0 : CGA memory (0B8000h) map (32K)
10864 0000F9F8 B800800B00          <1>     mov     eax, 0B8000h
10865 0000F9FD BB00800000          <1>     mov     ebx, 32768
10866 0000FA02 EB37                <1>     jmp     short sysvideo_26_3
10867                              <1> sysvideo_26_1:
10868                              <1>     ; BL = 1 : VGA memory (0A0000h) map (64K)
10869 0000FA04 B800000A00          <1>     mov     eax, 0A0000h
10870 0000FA09 BB00000100          <1>     mov     ebx, 65536
10871 0000FA0E EB2B                <1>     jmp     short sysvideo_26_3
10872                              <1> sysvideo_26_2:
10873                              <1>     ; BL = 2 : SVGA memory (LFB) map to user's buffer
10874 0000FA10 803D[86090000]02    <1>     cmp     byte [vbe3], 2  ; VESA VBE 2/3 vbios ready ?
```

```
10875 0000FA17 72AF              <1>     jb      short sysvideo_25  ; no, error !
10876 0000FA19 6681E200F0        <1>     and     dx, ~4095 ; clear low 12 bits
10877 0000FA1E 09D2              <1>     or      edx, edx  ; buffer size in bytes
10878 0000FA20 74A6              <1>     jz      short sysvideo_25  ; error
10879 0000FA22 89D3              <1>     mov     ebx, edx
10880 0000FA24 A1[FC9C0100]      <1>     mov     eax, [LFB_ADDR] ; [LFB_Info+LFBINFO.LFB_addr]
10881 0000FA29 21C0              <1>     and     eax, eax
10882 0000FA2B 7425              <1>     jz      short sysvideo_26_5
10883                            <1>                             ; (LFB parms are not set yet)
10884 0000FA2D 3B1D[009D0100]    <1>     cmp     ebx, [LFB_SIZE] ; [LFB_Info+LFBINFO.LFB_size]
10885 0000FA33 7606              <1>     jna     short sysvideo_26_3
10886 0000FA35 8B1D[009D0100]    <1>     mov     ebx, [LFB_SIZE]
10887                            <1> sysvideo_26_3:
10888 0000FA3B 52                <1>     push    edx
10889 0000FA3C 53                <1>     push    ebx     ; buffer size in bytes
10890 0000FA3D 51                <1>     push    ecx     ; user's buffer address
10891 0000FA3E 87CB              <1>     xchg    ebx, ecx
10892 0000FA40 C1E90C            <1>     shr     ecx, 12 ; convert buffer size to page count
10893 0000FA43 E87B65FFFF        <1>     call    direct_memory_access
10894 0000FA48 59                <1>     pop     ecx     ; user's buffer address
10895 0000FA49 5B                <1>     pop     ebx     ; buffer size
10896 0000FA4A 5A                <1>     pop     edx
10897                            <1>     ;jc     short sysvideo_25 ; error !
10898                            <1>                               ; [u.r0] = 0
10899                            <1>     ; 28/02/2021
10900 0000FA4B 7235              <1>     jc      short sysvideo_27_0 ; error !
10901                            <1>
10902                            <1> ;sysvideo_26_4:
10903                            <1>     ;mov    ebp, [u.usp] ; ebp points to user's registers
10904                            <1>     ;mov    [ebp+20], edx ; return to user with EDX value
10905                            <1>     ;mov    [ebp+16], ebx ; EBX
10906                            <1>     ;mov    [ebp+24], ecx ; ECX
10907                            <1>     ; eax = physical address of video memory (LFB)
10908                            <1>     ;mov    [u.r0], eax
10909                            <1>     ;jmp    sysret
10910 0000FA4D E92CFCFFFF        <1>     jmp     sysvideo_26_4
10911                            <1>
10912                            <1> sysvideo_26_5:
10913 0000FA52 66A1[1A0F0000]    <1>     mov     ax, [def_LFB_addr] ; default LFB for mode 118h
10914                            <1>     ; ah must be 0C0h or 0D0h or E0h
10915                            <1>     ; others are nonsence !?
10916 0000FA58 08E4              <1>     or      ah, ah
10917                            <1>     ;jz     short sysvideo_25 ; invalid lfb addr or
10918                            <1>                               ; it is not a vbe2 -bochs emu-
10919                            <1>                               ; or vbe3 -real- video bios
10920                            <1>     ; 28/02/2021
10921 0000FA5A 7426              <1>     jz      short sysvideo_27_0 ; invalid LFB address
10922                            <1>
10923 0000FA5C 80FCF0            <1>     cmp     ah, 0F0h
10924                            <1>     ;jnb    short sysvideo_25 ; nonsence !?
10925                            <1>     ; 28/02/2021
10926 0000FA5F 7321              <1>     jnb     short sysvideo_27_0 ; nonsence !?
10927                            <1>
10928 0000FA61 C1E010            <1>     shl     eax, 16
10929                            <1>     ;jz     short sysvideo_25 ; eax = 0
10930                            <1>
10931 0000FA64 81FB00907E00      <1>     cmp     ebx, 1920*1080*4 ; maximum value of possible
10932                            <1>                              ; buffer sizes
10933 0000FA6A 76CF              <1>     jna     short sysvideo_26_3  ; buffer size is proper
10934                            <1>     ; resize buffer to fit 4GB limit
10935 0000FA6C BB00907E00        <1>     mov     ebx, 1920*1080*4
10936 0000FA71 EBC8              <1>     jmp     short sysvideo_26_3
10937                            <1>
10938                            <1> sysvideo_27:
10939                            <1>     ; 23/07/2022
10940                            <1>     ; 16/02/2021
10941                            <1>     ; 18/01/2021
10942 0000FA73 80FF0C            <1>     cmp     bh, 12
10943                            <1>     ;ja     sysvideo_28 ; 19/01/2021
10944                            <1>     ; 23/07/2022
10945 0000FA76 7605              <1>     jna     short sysvideo_27_24
10946 0000FA78 E970010000        <1>     jmp     sysvideo_28
10947                            <1>
10948                            <1> sysvideo_27_24:    ; 23/07/2022
10949                            <1>     ; BH = 12
10950                            <1>     ; Font sub functions.
10951                            <1>     ; 12/02/2021
10952                            <1>     ; 11/01/2021
10953                            <1>     ; 10/01/2021
10954                            <1>     ;   BL = 0 : Disable system font overwrite
10955                            <1>     ;   BL = 1 : Enable system font overwrite
10956                            <1>     ;   BL = 2 : Read system font 8x8
10957                            <1>     ;   BL = 3 : Read system font 8x14
10958                            <1>     ;   BL = 4 : Read system font 8x16
10959                            <1>     ;   BL = 5 : Read user defined font 8x8
10960                            <1>     ;   BL = 6 : Read user defined font 8x16
10961                            <1>     ;   BL = 7 : Write system font 8x8
10962                            <1>     ;   BL = 8 : Write system font 8x14
10963                            <1>     ;   BL = 9 : Write system font 8x16
10964                            <1>     ;   BL = 10 : Write user defined font 8x8
10965                            <1>     ;   BL = 11 : Write user defined font 8x16
10966                            <1>     ;
10967                            <1>     ;  BL > 11 : invalid (not implemented)
10968                            <1>     ;
10969                            <1>     ; For BL = 1 to 11
10970                            <1>     ;   ECX = number of characters (<= 256)
10971                            <1>     ;   EDX = first character (ascii code in DL)
10972                            <1>     ;   ESI = user's buffer address
10973                            <1>     ;
10974                            <1>     ; Return: EAX = character count
10975                            <1>
10976 0000FA7D 80FB0B            <1>     cmp     bl, 11
10977 0000FA80 7605              <1>     jna     short sysvideo_27_1
10978                            <1> sysvideo_27_0:
10979 0000FA82 E9F7CFFFFF        <1>     jmp     sysret ; not implemented yet !
10980                            <1> sysvideo_27_1:
10981 0000FA87 66B80001          <1>     mov     ax, 256
10982 0000FA8B 08DB              <1>     or      bl, bl
10983 0000FA8D 750E              <1>     jnz     short sysvideo_27_3
10984                            <1>
10985                            <1>     ; bl = 0
10986                            <1>     ; disable system font overwrite
10987                            <1>
10988 0000FA8F 8025[4E9D0100]7F  <1>     and     byte [ufont], 7Fh ; clear bit 7
10989                            <1> sysvideo_27_2:
10990                            <1>     ;mov    word [u.r0], 256  ; > 0 -> successful
10991 0000FA96 A3[348E0100]      <1>     mov     [u.r0], eax ; 256
10992 0000FA9B EBE5              <1>     jmp     short sysvideo_27_0
10993                            <1> sysvideo_27_3:
10994 0000FA9D 80FB01            <1>     cmp     bl, 1
10995 0000FAA0 7710              <1>     ja      short sysvideo_27_4
10996                            <1>
10997                            <1>     ; bl = 1
10998                            <1>     ; enable system font overwrite
```

```
10999                                   <1>     ;       if [multi_tasking]= 0 and [u.uid] = 0
11000                                   <1>
11001                                   <1>     ;cmp    byte [multi_tasking], 0
11002                                   <1>     ;                       ; multi tasking enabled ?
11003                                   <1>     ;ja     short sysvideo_27_0 ; yes
11004                                   <1>     ;; 19/01/2021
11005                                   <1>     ;; system maintenance or single user mode
11006                                   <1>     ;cmp    byte [u.uid], 0  ; root ?
11007                                   <1>     ;ja     short sysvideo_27_0 ; no
11008                                   <1>
11009                                   <1>     ; 19/01/2021
11010                                   <1>     ; multi tasking & root check
11011 0000FAA2 E8EBFEFFFF              <1>     call    sysvideo_22_4
11012 0000FAA7 73D9                    <1>     jnc     short sysvideo_27_0 ; not permitted
11013                                   <1>
11014                                   <1>     ; [multi_tasking]= 0 and [u.uid] = 0
11015                                   <1>
11016 0000FAA9 800D[4E9D0100]80        <1>     or      byte [ufont], 80h ; set bit 7
11017                                   <1>
11018 0000FAB0 EBE4                    <1>     jmp     short sysvideo_27_2
11019                                   <1>
11020                                   <1> sysvideo_27_4:
11021 0000FAB2 09C9                    <1>     or      ecx, ecx
11022 0000FAB4 74CC                    <1>     jz      short sysvideo_27_0
11023 0000FAB6 21D2                    <1>     and     edx, edx
11024 0000FAB8 7410                    <1>     jz      short sysvideo_27_4_0
11025                                   <1>     ;mov    ax, 256
11026 0000FABA 39C1                    <1>     cmp     ecx, eax ; 256
11027 0000FABC 77C4                    <1>     ja      short sysvideo_27_0
11028 0000FABE 48                      <1>     dec     eax
11029 0000FABF 39C2                    <1>     cmp     edx, eax ; 255
11030 0000FAC1 77BF                    <1>     ja      short sysvideo_27_0
11031 0000FAC3 40                      <1>     inc     eax
11032 0000FAC4 29D0                    <1>     sub     eax, edx ; 256 - DX
11033 0000FAC6 39C8                    <1>     cmp     eax, ecx
11034 0000FAC8 72B8                    <1>     jb      short sysvideo_27_0
11035                                   <1>
11036                                   <1> sysvideo_27_4_0:
11037 0000FACA 89F5                    <1>     mov     ebp, esi
11038                                   <1>
11039 0000FACC 80FB06                  <1>     cmp     bl, 6
11040 0000FACF 7768                    <1>     ja      short sysvideo_27_13
11041 0000FAD1 7210                    <1>     jb      short sysvideo_27_5
11042                                   <1>     ; bl = 6
11043 0000FAD3 F605[4E9D0100]10        <1>     test    byte [ufont], 16 ; 8x16 user font loaded ?
11044 0000FADA 74A6                    <1>     jz      short sysvideo_27_0
11045                                   <1>     ; read 8x16 user defined font
11046 0000FADC BE00400900              <1>     mov     esi, VGAFONT16USER
11047 0000FAE1 EB0C                    <1>     jmp     short sysvideo_27_6
11048                                   <1> sysvideo_27_5:
11049 0000FAE3 80FB04                  <1>     cmp     bl, 4
11050 0000FAE6 7239                    <1>     jb      short sysvideo_27_11
11051 0000FAE8 7721                    <1>     ja      short sysvideo_27_9
11052                                   <1>     ; bl = 4
11053                                   <1>     ; read 8x16 system font
11054 0000FAEA BE[10630100]            <1>     mov     esi, vgafont16
11055                                   <1> sysvideo_27_6:
11056                                   <1>     ; read 8x16 font
11057                                   <1>     ;shl    dx, 4 ; * 16
11058                                   <1>     ;shl    cx, 4 ; * 16 ; 16 bytes per char
11059                                   <1>     ; 23/07/2022
11060 0000FAEF C1E204                  <1>     shl     edx, 4 ; * 16
11061 0000FAF2 C1E104                  <1>     shl     ecx, 4 ; * 16
11062                                   <1> sysvideo_27_7:
11063 0000FAF5 89EF                    <1>     mov     edi, ebp
11064                                   <1>     ;add    edi, edx ; 16/02/2021
11065 0000FAF7 01D6                    <1>     add     esi, edx
11066                                   <1>     ; ecx = byte count
11067                                   <1>     ; esi = source (in system memory)
11068                                   <1>     ; edi = destination (in user memory)
11069 0000FAF9 E8EC0F0000              <1>     call    transfer_to_user_buffer
11070 0000FAFE 7206                    <1>     jc      short sysvideo_27_8
11071 0000FB00 890D[348E0100]          <1>     mov     [u.r0], ecx
11072                                   <1> sysvideo_27_8:
11073 0000FB06 E973CFFFFF              <1>     jmp     sysret
11074                                   <1> sysvideo_27_9:
11075                                   <1>     ; bl = 5
11076 0000FB0B F605[4E9D0100]08        <1>     test    byte [ufont], 8 ; 8x8 user font loaded ?
11077 0000FB12 74F2                    <1>     jz      short sysvideo_27_8
11078                                   <1>     ; read 8x8 user defined font
11079 0000FB14 BE00500900              <1>     mov     esi, VGAFONT8USER
11080                                   <1> sysvideo_27_10:
11081                                   <1>     ; read 8x8 font
11082                                   <1>     ;shl    dx, 3 ; * 8
11083                                   <1>     ;shl    cx, 3 ; * 8 ; 8 bytes per char
11084                                   <1>     ; 23/07/2022
11085 0000FB19 C1E203                  <1>     shl     edx, 3 ; * 8
11086 0000FB1C C1E103                  <1>     shl     ecx, 3 ; * 8
11087 0000FB1F EBD4                    <1>     jmp     short sysvideo_27_7
11088                                   <1>
11089                                   <1> sysvideo_27_11:
11090 0000FB21 80FB03                  <1>     cmp     bl, 3  ; 8x14 system font
11091 0000FB24 720C                    <1>     jb      short sysvideo_27_12 ; 8x8 system font
11092                                   <1>     ; bl = 3
11093                                   <1>     ; read 8x14 system font
11094                                   <1>     ;mov    al, 14
11095                                   <1>     ;mul    dl
11096                                   <1>     ;mov    dx, ax
11097                                   <1>     ;push   edx
11098                                   <1>     ;mov    ax, 14
11099                                   <1>     ;mul    cx
11100                                   <1>     ;mov    cx, ax
11101                                   <1>     ;pop    edx
11102 0000FB26 E8A5000000              <1>     call    sysvideo_27_14
11103 0000FB2B BE[10550100]            <1>     mov     esi, vgafont14
11104 0000FB30 EBC3                    <1>     jmp     short sysvideo_27_7
11105                                   <1>
11106                                   <1> sysvideo_27_12:
11107                                   <1>     ; bl = 2
11108                                   <1>     ; read 8x8 system font
11109 0000FB32 BE[104D0100]            <1>     mov     esi, vgafont8
11110 0000FB37 EBE0                    <1>     jmp     short sysvideo_27_10
11111                                   <1>
11112                                   <1> sysvideo_27_13:
11113                                   <1>     ; overwrite font
11114 0000FB39 80FB0A                  <1>     cmp     bl, 10
11115 0000FB3C 776E                    <1>     ja      short sysvideo_27_22 ; 8x16 user font
11116 0000FB3E 7224                    <1>     jb      short sysvideo_27_15
11117                                   <1>     ; bl = 10
11118 0000FB40 BF00500900              <1>     mov     edi, VGAFONT8USER
11119 0000FB45 F605[4E9D0100]08        <1>     test    byte [ufont], 8 ; 8x8 user font loaded ?
11120 0000FB4C 7556                    <1>     jnz     short sysvideo_27_21 ; yes
11121 0000FB4E 08ED                    <1>     or      ch, ch ; cx = 256
11122                                   <1>     ;jnz    short sysvideo_27_21 ; 256 chars
```

```
11123 0000FB50 7406                <1>     jz      short sysvideo_27_13_0
11124 0000FB52 66B90008            <1>     mov     cx, 8*256
11125 0000FB56 EB35                <1>     jmp     short sysvideo_27_18_0
11126                              <1> sysvideo_27_13_0:
11127                              <1>     ; copy system font to user font before overwrite
11128 0000FB58 BE[104D0100]        <1>     mov     esi, vgafont8
11129                              <1>     ;push   edi
11130                              <1>     ;push   ecx
11131                              <1>     ;mov    cl, 64
11132                              <1>     ;rep    movsd
11133                              <1>     ;pop    ecx
11134                              <1>     ;pop    edi
11135                              <1>     ;mov    esi, ebp ; user's font buffer
11136 0000FB5D E880000000          <1>     call    sysvideo_27_23
11137 0000FB62 EB40                <1>     jmp     short sysvideo_27_21
11138                              <1>
11139                              <1> sysvideo_27_15:
11140                              <1>     ; check system font overwrite permission
11141 0000FB64 F605[4E9D0100]80    <1>     test    byte [ufont], 80h
11142 0000FB6B 7499                <1>     jz      short sysvideo_27_8
11143                              <1>
11144 0000FB6D 80FB08              <1>     cmp     bl, 8
11145 0000FB70 773A                <1>     ja      short sysvideo_27_22 ; 8x16 system font
11146 0000FB72 722B                <1>     jb      short sysvideo_27_20 ; 8x8 system font
11147                              <1>     ; bl = 8
11148                              <1>     ; overwrite 8x14 system font
11149                              <1>     ;mov    al, 14
11150                              <1>     ;mul    dl
11151                              <1>     ;mov    dx, ax
11152                              <1>     ;push   edx
11153                              <1>     ;mov    ax, 14
11154                              <1>     ;mul    cx
11155                              <1>     ;mov    cx, ax
11156                              <1>     ;pop    edx
11157 0000FB74 E857000000          <1>     call    sysvideo_27_14
11158 0000FB79 BF[10550100]        <1>     mov     edi, vgafont14
11159 0000FB7E EB0B                <1>     jmp     short sysvideo_27_18
11160                              <1> sysvideo_27_16:
11161                              <1>     ; bl = 9
11162                              <1>     ; overwrite 8x16 system font
11163 0000FB80 BF[10630100]        <1>     mov     edi, vgafont16
11164                              <1> sysvideo_27_17:
11165                              <1>     ; overwrite 8x16 font
11166                              <1>     ;shl    dx, 4 ; * 16
11167                              <1>     ;shl    cx, 4 ; * 16 ; 16 bytes per char
11168                              <1>     ; 23/07/2022
11169 0000FB85 C1E204              <1>     shl     edx, 4 ; * 16
11170 0000FB88 C1E104              <1>     shl     ecx, 4 ; * 16
11171                              <1> sysvideo_27_18:
11172 0000FB8B 01D7                <1>     add     edi, edx
11173                              <1>     ;add    esi, edx ; 16/02/2021
11174                              <1> sysvideo_27_18_0:
11175                              <1>     ; ecx = byte count
11176                              <1>     ; esi = source (in user memory)
11177                              <1>     ; edi = destination (in system memory)
11178 0000FB8D E8A20F0000          <1>     call    transfer_from_user_buffer
11179 0000FB92 7206                <1>     jc      short sysvideo_27_19
11180 0000FB94 890D[348E0100]      <1>     mov     [u.r0], ecx
11181                              <1> sysvideo_27_19:
11182 0000FB9A E9DFCEFFFF          <1>     jmp     sysret
11183                              <1> sysvideo_27_20:
11184                              <1>     ; bl = 7
11185                              <1>     ; overwrite 8x8 system font
11186 0000FB9F BF[104D0100]        <1>     mov     edi, vgafont8
11187                              <1> sysvideo_27_21:
11188                              <1>     ; overwrite 8x8 font
11189                              <1>     ;shl    dx, 3 ; * 8
11190                              <1>     ;shl    cx, 3 ; * 8 ; 8 bytes per char
11191                              <1>     ; 23/07/2022
11192 0000FBA4 C1E203              <1>     shl     edx, 3 ; * 8
11193 0000FBA7 C1E103              <1>     shl     ecx, 3 ; * 8
11194 0000FBAA EBDF                <1>     jmp     short sysvideo_27_18
11195                              <1> sysvideo_27_22:
11196                              <1>     ; bl = 11
11197                              <1>     ; overwrite 8x16 user defined font
11198 0000FBAC BF00400900          <1>     mov     edi, VGAFONT16USER
11199 0000FBB1 F605[4E9D0100]10    <1>     test    byte [ufont], 16 ; 8x16 user font loaded ?
11200 0000FBB8 75CB                <1>     jnz     short sysvideo_27_17 ; yes
11201 0000FBBA 08ED                <1>     or      ch, ch ; cx = 256
11202                              <1>     ;jnz    short sysvideo_27_17 ; 256 chars
11203 0000FBBC 7406                <1>     jz      short sysvideo_27_22_0
11204 0000FBBE 66B90010            <1>     mov     cx, 16*256
11205 0000FBC2 EBC9                <1>     jmp     short sysvideo_27_18_0
11206                              <1> sysvideo_27_22_0:
11207                              <1>     ; copy system font to user font before overwrite
11208 0000FBC4 BE[10630100]        <1>     mov     esi, vgafont16
11209                              <1>     ;push   edi
11210                              <1>     ;push   ecx
11211                              <1>     ;mov    cl, 64
11212                              <1>     ;rep    movsd
11213                              <1>     ;pop    ecx
11214                              <1>     ;pop    edi
11215                              <1>     ;mov    esi, ebp ; user's font buffer
11216 0000FBC9 E814000000          <1>     call    sysvideo_27_23
11217 0000FBCE EBB5                <1>     jmp     short sysvideo_27_17
11218                              <1>
11219                              <1> sysvideo_27_14:
11220                              <1>     ; 16/02/2021
11221 0000FBD0 52                  <1>     push    edx
11222 0000FBD1 66B80E00            <1>     mov     ax, 14
11223 0000FBD5 66F7E1              <1>     mul     cx
11224 0000FBD8 89C1                <1>     mov     ecx, eax
11225 0000FBDA 5A                  <1>     pop     edx
11226 0000FBDB B00E                <1>     mov     al, 14
11227 0000FBDD F6E2                <1>     mul     dl
11228 0000FBDF 89C2                <1>     mov     edx, eax
11229 0000FBE1 C3                  <1>     retn
11230                              <1>
11231                              <1>     ;mov    al, 14
11232                              <1>     ;mul    dl
11233                              <1>     ;mov    dx, ax
11234                              <1>     ;push   edx
11235                              <1>     ;; 12/02/2021
11236                              <1>     ;mov    ax, 14
11237                              <1>     ;;mov   eax, 14
11238                              <1>     ;;mul   cx
11239                              <1>     ;mul    ecx
11240                              <1>     ;;mov   cx, ax
11241                              <1>     ;mov    ecx, eax
11242                              <1>     ;pop    edx
11243                              <1>     ;retn
11244                              <1>
11245                              <1> sysvideo_27_23:
11246 0000FBE2 57                  <1>     push    edi
```

```
11247 0000FBE3 51                <1>     push    ecx
11248 0000FBE4 B140              <1>     mov     cl, 64
11249 0000FBE6 F3A5              <1>     rep     movsd
11250 0000FBE8 59                <1>     pop     ecx
11251 0000FBE9 5F                <1>     pop     edi
11252 0000FBEA 89EE              <1>     mov     esi, ebp ; user's font buffer
11253 0000FBEC C3                <1>     retn
11254                            <1>
11255                            <1> sysvideo_28:
11256                            <1>     ; 23/07/2022
11257                            <1>     ; 24/01/2021
11258                            <1>     ; 23/01/2021
11259                            <1>     ; 18/01/2021
11260 0000FBED 80FF0E            <1>     cmp     bh, 14
11261                            <1>     ;jb     sysvideo_29
11262                            <1>     ; 23/07/2022
11263 0000FBF0 7222              <1>     jb      short sysvideo_28_29
11264                            <1>     ;ja     sysvideo_30
11265                            <1>     ; 23/07/2022
11266 0000FBF2 7725              <1>     ja      short sysvideo_28_30
11267                            <1>
11268                            <1>     ; BH = 14
11269                            <1>     ; Save/Restore Super VGA video state
11270                            <1>
11271                            <1>     ; BL = options
11272                            <1>     ;       bit 0 - Save (0) or Restore (1)
11273                            <1>     ;       bit 1 - controller hardware state
11274                            <1>     ;       bit 2 - BIOS data state
11275                            <1>     ;       bit 3 - DAC state
11276                            <1>     ;       bit 4 - (extended) Register state
11277                            <1>     ;       bit 5 - system (0) or user (1) memory
11278                            <1>     ;       bit 6 - verify without transfer
11279                            <1>     ;       bit 7 - not used (must be 0)
11280                            <1>
11281                            <1>     ; ECX = Buffer address or VideoStateID
11282                            <1>
11283 0000FBF4 803D[86090000]02  <1>     cmp     byte [vbe3], 2 ;  VESA VBE2 or VBE3 ?
11284 0000FBFB 7721              <1>     ja      short sysvideo_28_0 ; yes
11285 0000FBFD 7210              <1>     jb      short sysvideo_28_16 ; not a SVGA sys !
11286                            <1>
11287                            <1>     ; == VBE2 ==
11288                            <1>     ; Check Bochs/Qemu/VirtualBox PC emulator
11289                            <1>     ; (vbe2 is usable only for emulator's vbios)
11290 0000FBFF 8A25[87090000]    <1>     mov     ah, [vbe2bios]
11291 0000FC05 80FCC0            <1>     cmp     ah, 0C0h
11292 0000FC08 7205              <1>     jb      short sysvideo_28_16 ; unknown vbios !
11293 0000FC0A 80FCC5            <1>     cmp     ah, 0C5h
11294 0000FC0D 760F              <1>     jna     short sysvideo_28_0
11295                            <1>             ; Use kernel's vbios functions (video.s)
11296                            <1> sysvideo_28_16:
11297                            <1>     ; unknown vbios !
11298 0000FC0F E96ACEFFFF        <1>     jmp     sysret
11299                            <1>
11300                            <1> sysvideo_28_29:
11301                            <1>     ; 23/07/2022
11302 0000FC14 E955010000        <1>     jmp     sysvideo_29
11303                            <1> sysvideo_28_30:
11304                            <1>     ; 23/07/2022
11305 0000FC19 E935020000        <1>     jmp     sysvideo_30
11306                            <1>
11307                            <1> sysvideo_28_0:
11308 0000FC1E 80FB7F            <1>     cmp     bl, 7Fh
11309 0000FC21 77EC              <1>     ja      short sysvideo_28_16 ; unknown options
11310                            <1>
11311 0000FC23 88DA              <1>     mov     dl, bl
11312 0000FC25 80E21F            <1>     and     dl, 1Fh
11313 0000FC28 D0EA              <1>     shr     dl, 1
11314 0000FC2A 74E3              <1>     jz      short sysvideo_28_16 ; invalid !
11315                            <1>     ; DL = VBE Function 4F04h Save/Restore options
11316                            <1>     ;  bit 0 : controller hardware state
11317                            <1>     ;  bit 1 : BIOS data state
11318                            <1>     ;  bit 2 : DAC state
11319                            <1>     ;  bit 3 : (extended) Register state
11320                            <1>
11321 0000FC2C F6C320            <1>     test    bl, 32 ; bit 5
11322                            <1>     ;jnz    sysvideo_28_7 ; user buffer
11323                            <1>     ; 23/07/2022
11324 0000FC2F 7405              <1>     jz      short sysvideo_28_17
11325 0000FC31 E9B1000000        <1>     jmp     sysvideo_28_7
11326                            <1>
11327                            <1> sysvideo_28_17:     ; 23/07/2022
11328                            <1>     ; source or destination is kernel/system buffer
11329                            <1>
11330 0000FC36 803D[509D0100]00  <1>     cmp     byte [srvsf], 0 ; srs permission flag
11331 0000FC3D 76D0              <1>     jna     short sysvideo_28_16 ; not permitted
11332                            <1>
11333 0000FC3F F6C301            <1>     test    bl, 1
11334 0000FC42 743A              <1>     jz      short sysvideo_28_4 ; Save
11335                            <1>
11336                            <1>     ; Restore
11337 0000FC44 3B0D[529D0100]    <1>     cmp     ecx, [VideoStateID]
11338 0000FC4A 75C3              <1>     jne     short sysvideo_28_16 ; not correct ID !
11339                            <1>
11340 0000FC4C 0FB6CA            <1>     movzx   ecx, dl
11341 0000FC4F 80CA80            <1>     or      dl, 80h
11342 0000FC52 3A15[519D0100]    <1>     cmp     dl, [srvso]
11343 0000FC58 75B5              <1>     jne     short sysvideo_28_16 ; not correct !
11344                            <1>
11345 0000FC5A 88DA              <1>     mov     dl, bl
11346                            <1>
11347                            <1>     ; ecx = cl = options
11348 0000FC5C E8163FFFFF        <1>     call    vbe_srs_gbs
11349                            <1>     ; ebx = state buffer size (data size)
11350                            <1>
11351 0000FC61 891D[348E0100]    <1>     mov     [u.r0], ebx
11352                            <1>
11353 0000FC67 F6C240            <1>     test    dl, 64 ; verify without transfer
11354 0000FC6A 75A3              <1>     jnz     short sysvideo_28_16 ; yes
11355                            <1>
11356 0000FC6C BE00580900        <1>     mov     esi, VBE3VIDEOSTATE
11357 0000FC71 BF00760900        <1>     mov     edi, VBE3SAVERESTOREBLOCK
11358 0000FC76 87D9              <1>     xchg    ecx, ebx
11359 0000FC78 F3A4              <1>     rep     movsb
11360                            <1>
11361 0000FC7A 88D9              <1>     mov     cl, bl
11362                            <1>
11363                            <1>     ; 23/01/2021
11364 0000FC7C EB44              <1>     jmp     short sysvideo_28_10
11365                            <1>
11366                            <1> sysvideo_28_4:
11367 0000FC7E 53                <1>     push    ebx
11368                            <1>     ; 24/01/2021
11369 0000FC7F 31DB              <1>     xor     ebx, ebx ; 0 ; use kernel's buffer
11370 0000FC81 881D[519D0100]    <1>     mov     [srvso], bl ; 0 ; invalidate
```

```
11371 0000FC87 891D[529D0100]      <1>     mov     [VideoStateID], ebx ; 0 ; invalidate
11372 0000FC8D 0FB6CA              <1>     movzx   ecx, dl ; options
11373 0000FC90 B201                <1>     mov     dl, 1 ; save state
11374 0000FC92 E890000000          <1>     call    sysvideo_28_11 ; 23/01/2021
11375                              <1>     ; Note: VBE3 BIOS data save option will be
11376                              <1>     ;       disabled.. ; 24/01/2021
11377 0000FC97 89CA                <1>     mov     edx, ecx ; state (save) options
11378 0000FC99 5B                  <1>     pop     ebx
11379                              <1>
11380 0000FC9A 6683F84F            <1>     cmp     ax, 4Fh ; successful ?
11381 0000FC9E 7536                <1>     jne     short sysvideo_28_3 ; no !
11382                              <1>
11383 0000FCA0 F6C340              <1>     test    bl, 64 ; verify without transfer
11384 0000FCA3 7536                <1>     jnz     short sysvideo_28_6 ; yes
11385                              <1>
11386                              <1>     ; ecx = cl = options
11387 0000FCA5 E8CD3EFFFF          <1>     call    vbe_srs_gbs
11388                              <1>     ; ebx = state buffer size (data size)
11389                              <1>
11390 0000FCAA BE00760900          <1>     mov     esi, VBE3SAVERESTOREBLOCK
11391 0000FCAF BF00580900          <1>     mov     edi, VBE3VIDEOSTATE
11392 0000FCB4 89D9                <1>     mov     ecx, ebx
11393 0000FCB6 F3A4                <1>     rep     movsb
11394                              <1>
11395 0000FCB8 88D1                <1>     mov     cl, dl
11396 0000FCBA 80C980              <1>     or      cl, 80h ; SVGA (VESA VBE) flag
11397                              <1>     ;mov    [srvso], dl
11398                              <1>
11399 0000FCBD E908010000          <1>     jmp     sysvideo_28_15
11400                              <1>
11401                              <1>     ; 23/01/2021
11402                              <1> sysvideo_28_10:
11403                              <1>     ; CL = VESA VBE3 Save/Restore options
11404                              <1>
11405 0000FCC2 B202                <1>     mov     dl, 2 ; restore state
11406                              <1>
11407 0000FCC4 E85C000000          <1>     call    sysvideo_28_1
11408                              <1>
11409 0000FCC9 6683F84F            <1>     cmp     ax, 4Fh ; successful ?
11410 0000FCCD 7407                <1>     je      short sysvideo_28_3
11411                              <1>     ;jmp    short sysvideo_28_9
11412                              <1>
11413                              <1> sysvideo_28_9:
11414                              <1>     ; return zero size (error) to user
11415 0000FCCF 29C0                <1>     sub     eax, eax
11416                              <1> sysvideo_28_5:
11417 0000FCD1 A3[348E0100]        <1>     mov     [u.r0], eax
11418                              <1> sysvideo_28_3:
11419 0000FCD6 E9A3CDFFFF          <1>     jmp     sysret
11420                              <1>
11421                              <1> sysvideo_28_6:
11422                              <1>     ; use timer ticks as VideoStateID
11423 0000FCDB A1[00770100]        <1>     mov     eax, [TIMER_LH]
11424 0000FCE0 09C0                <1>     or      eax, eax
11425 0000FCE2 75ED                <1>     jnz     short sysvideo_28_5
11426 0000FCE4 40                  <1>     inc     eax
11427 0000FCE5 EBEA                <1>     jmp     short sysvideo_28_5
11428                              <1>
11429                              <1> sysvideo_28_7:
11430                              <1>     ; save/restore to/from user buffer
11431                              <1>
11432                              <1>     ; 23/01/2021
11433 0000FCE7 89CE                <1>     mov     esi, ecx ; user's vstate buffer
11434 0000FCE9 BF00760900          <1>     mov     edi, VBE3SAVERESTOREBLOCK
11435                              <1>
11436 0000FCEE 0FB6CA              <1>     movzx   ecx, dl ; VESA VBE func 4F04h options
11437                              <1>
11438                              <1>     ; source or destination is user buffer
11439 0000FCF1 F6C301              <1>     test    bl, 1
11440 0000FCF4 7444                <1>     jz      short sysvideo_28_12  ; Save
11441                              <1>
11442                              <1>     ; Restore
11443 0000FCF6 803D[509D0100]00    <1>     cmp     byte [srvsf], 0 ; srs permission flag
11444 0000FCFD 766A                <1>     jna     short sysvideo_28_14 ; not permitted
11445                              <1>
11446 0000FCFF 88DA                <1>     mov     dl, bl ; 'sysvideo' options
11447                              <1>
11448                              <1>     ; ecx = cl = options
11449 0000FD01 E8713EFFFF          <1>     call    vbe_srs_gbs
11450                              <1>     ; ebx = state buffer size (data size)
11451                              <1>
11452 0000FD06 891D[348E0100]      <1>     mov     [u.r0], ebx ; transfer count
11453                              <1>
11454 0000FD0C F6C240              <1>     test    dl, 64 ; verify without transfer
11455 0000FD0F 7558                <1>     jnz     short sysvideo_28_14 ; yes
11456                              <1>
11457 0000FD11 6681FB0008          <1>     cmp     bx, 2048
11458 0000FD16 73B7                <1>     jnb     short sysvideo_28_9 ; invalid
11459                              <1>
11460 0000FD18 87D9                <1>     xchg    ecx, ebx
11461                              <1>     ; esi = user buffer
11462                              <1>     ; edi = VBE3SAVERESTOREBLOCK
11463                              <1>
11464 0000FD1A E8150E0000          <1>     call    transfer_from_user_buffer
11465 0000FD1F 72AE                <1>     jc      short sysvideo_28_9 ; error
11466                              <1>
11467 0000FD21 89D9                <1>     mov     ecx, ebx ; Function 4F04h options
11468 0000FD23 EB9D                <1>     jmp     short sysvideo_28_10 ; 23/01/2021
11469                              <1>
11470                              <1> sysvideo_28_1:
11471 0000FD25 31DB                <1>     xor     ebx, ebx ; 0 ; use kernel's buffer
11472                              <1> sysvideo_28_11:
11473                              <1>     ; 24/01/2021
11474 0000FD27 803D[86090000]03    <1>     cmp     byte [vbe3], 3
11475 0000FD2E 7405                <1>     je      short sysvideo_28_2
11476                              <1>
11477                              <1>     ; VESA VBE2 (BOCHS/QEMU/VBOX) video bios
11478 0000FD30 E9AB3DFFFF          <1>     jmp     _vbe_biosfn_save_restore_state
11479                              <1> sysvideo_28_2:
11480                              <1>     ;24/01/2021
11481                              <1>     ;mov    eax, 4F04h  ; Save/Restore vstate
11482                              <1>     ; VESA VBE3 video bios
11483 0000FD35 E9211DFFFF          <1>     jmp     _vbe3_pmfn_save_restore_state
11484                              <1>
11485                              <1> sysvideo_28_12:
11486                              <1>     ; Save
11487                              <1>     ;mov    edi, VBE3SAVERESTOREBLOCK
11488                              <1>
11489                              <1>     ;movzx  ecx, dl ; options
11490 0000FD3A 56                  <1>     push    esi
11491 0000FD3B 53                  <1>     push    ebx
11492                              <1>     ; 23/01/2021
11493 0000FD3C B201                <1>     mov     dl, 1 ; save state
11494 0000FD3E E8E2FFFFFF          <1>     call    sysvideo_28_1
```

```
11495 0000FD43 5A                  <1>     pop     edx ; 'sysvideo' options
11496 0000FD44 5F                  <1>     pop     edi ; user's video state buffer
11497                              <1>
11498 0000FD45 6683F84F            <1>     cmp     ax, 4Fh ; successful ?
11499 0000FD49 751E                <1>     jne     short sysvideo_28_14 ; no !
11500                              <1>
11501                              <1>     ; ecx = cl = options
11502 0000FD4B E8273EFFFF          <1>     call    vbe_srs_gbs
11503                              <1>     ; ebx = state buffer size (data size)
11504                              <1>
11505 0000FD50 89D9                <1>     mov     ecx, ebx ; transfer count
11506                              <1>
11507 0000FD52 F6C240              <1>     test    dl, 64 ; verify without transfer
11508 0000FD55 750C                <1>     jnz     short sysvideo_28_13 ; yes
11509                              <1>
11510                              <1>     ;mov    edi, esi
11511 0000FD57 BE00760900          <1>     mov     esi, VBE3SAVERESTOREBLOCK
11512 0000FD5C E8890D0000          <1>     call    transfer_to_user_buffer
11513 0000FD61 7206                <1>     jc      short sysvideo_28_14
11514                              <1> sysvideo_28_13:
11515 0000FD63 890D[348E0100]      <1>     mov     [u.r0], ecx
11516                              <1> sysvideo_28_14:
11517 0000FD69 E910CDFFFF          <1>     jmp     sysret
11518                              <1>
11519                              <1> sysvideo_29:
11520                              <1>     ; 18/01/2021
11521                              <1>     ; BH = 13
11522                              <1>     ; Save/Restore std VGA video state
11523                              <1>
11524                              <1>     ; bl = 0..3
11525                              <1>     ;       save to or restore from
11526                              <1>     ;       system buffer, VBE3VIDEOSTATE
11527                              <1>     ;       ECX = VideoStateID for restoring
11528                              <1>     ; bl = 4..7
11529                              <1>     ;       save to or restore from
11530                              <1>     ;       user buffer pointed by ECX
11531                              <1>
11532 0000FD6E 80FB03              <1>     cmp     bl, 3
11533 0000FD71 7776                <1>     ja      short sysvideo_29_6
11534                              <1>
11535                              <1>     ; source or destination is kernel/system buffer
11536                              <1>
11537 0000FD73 803D[509D0100]00    <1>     cmp     byte [srvsf], 0 ; srs permission flag
11538 0000FD7A 7668                <1>     jna     short sysvideo_29_5 ; not permitted
11539                              <1>
11540 0000FD7C F6C301              <1>     test    bl, 1
11541 0000FD7F 7437                <1>     jz      short sysvideo_29_2  ; Save
11542                              <1>
11543                              <1>     ; Restore
11544 0000FD81 3B0D[529D0100]      <1>     cmp     ecx, [VideoStateID]
11545 0000FD87 755B                <1>     jne     short sysvideo_29_5 ; not correct ID !
11546 0000FD89 80FB01              <1>     cmp     bl, 1
11547 0000FD8C 7709                <1>     ja      short sysvideo_29_0
11548                              <1>     ; bl = 1
11549 0000FD8E BB6E000000          <1>     mov     ebx, 110
11550 0000FD93 B103                <1>     mov     cl, 3 ; ctrl, vbios data
11551 0000FD95 EB07                <1>     jmp     short sysvideo_29_1
11552                              <1> sysvideo_29_0:
11553                              <1>     ; bl  = 3
11554 0000FD97 BB72030000          <1>     mov     ebx, 882
11555 0000FD9C B107                <1>     mov     cl, 7 ; ctrl, vbios data, dac
11556                              <1> sysvideo_29_1:
11557 0000FD9E 3A0D[519D0100]      <1>     cmp     cl, [srvso]
11558 0000FDA4 753E                <1>     jne     short sysvideo_29_5 ; not correct !
11559                              <1>
11560 0000FDA6 BE00580900          <1>     mov     esi, VBE3VIDEOSTATE ; 22/01/2021
11561 0000FDAB E85B3FFFFF          <1>     call    biosfn_restore_video_state
11562 0000FDB0 891D[348E0100]      <1>     mov     [u.r0], ebx ; video state size (bytes)
11563                              <1>     ;jmp    sysret
11564 0000FDB6 EB2C                <1>     jmp     short sysvideo_29_5
11565                              <1> sysvideo_29_2:
11566                              <1>     ;mov    esi, ecx
11567 0000FDB8 BF00580900          <1>     mov     edi, VBE3VIDEOSTATE
11568                              <1>
11569 0000FDBD B107                <1>     mov     cl, 7 ; ctrl, vbios data, dac
11570 0000FDBF 08DB                <1>     or      bl, bl
11571 0000FDC1 7502                <1>     jnz     short sysvideo_29_3 ; bl = 2
11572                              <1>     ; bl = 0
11573 0000FDC3 B103                <1>     mov     cl, 3 ; ctrl, vbios data
11574                              <1> sysvideo_29_3:
11575 0000FDC5 E8D93DFFFF          <1>     call    biosfn_save_video_state
11576                              <1> sysvideo_28_15:
11577                              <1>     ; use timer ticks as VideoStateID
11578 0000FDCA A1[00770100]        <1>     mov     eax, [TIMER_LH]
11579 0000FDCF 21C0                <1>     and     eax, eax
11580 0000FDD1 7501                <1>     jnz     short sysvideo_29_4
11581 0000FDD3 40                  <1>     inc     eax
11582                              <1> sysvideo_29_4:
11583 0000FDD4 880D[519D0100]      <1>     mov     [srvso], cl
11584 0000FDDA A3[529D0100]        <1>     mov     [VideoStateID], eax
11585 0000FDDF A3[348E0100]        <1>     mov     [u.r0], eax
11586                              <1> sysvideo_29_5:
11587 0000FDE4 E995CCFFFF          <1>     jmp     sysret
11588                              <1>
11589                              <1> sysvideo_29_6:
11590 0000FDE9 80FB07              <1>     cmp     bl, 7
11591 0000FDEC 77F6                <1>     ja      short sysvideo_29_5 ; invalid sub function
11592                              <1>
11593 0000FDEE 89CE                <1>     mov     esi, ecx
11594 0000FDF0 BF00760900          <1>     mov     edi, VBE3SAVERESTOREBLOCK
11595                              <1>
11596                              <1>     ; source or destination is user buffer
11597 0000FDF5 F6C301              <1>     test    bl, 1
11598 0000FDF8 7434                <1>     jz      short sysvideo_29_9  ; Save
11599                              <1>
11600                              <1>     ; Restore
11601 0000FDFA 803D[509D0100]00    <1>     cmp     byte [srvsf], 0 ; srs permission flag
11602 0000FE01 76E1                <1>     jna     short sysvideo_29_5 ; not permitted
11603                              <1>
11604                              <1>     ;mov    esi, ecx
11605                              <1>     ;mov    edi, VBE3SAVERESTOREBLOCK
11606                              <1>
11607 0000FE03 80FB07              <1>     cmp     bl, 7
11608 0000FE06 7409                <1>     je      short sysvideo_29_7
11609                              <1>     ; bl = 5
11610 0000FE08 B303                <1>     mov     bl, 3
11611 0000FE0A B96E000000          <1>     mov     ecx, 110
11612 0000FE0F EB05                <1>     jmp     short sysvideo_29_8
11613                              <1> sysvideo_29_7:
11614                              <1>     ; bl = 7
11615 0000FE11 B972030000          <1>     mov     ecx, 882
11616                              <1> sysvideo_29_8:
11617 0000FE16 E8190D0000          <1>     call    transfer_from_user_buffer
11618 0000FE1B 72C7                <1>     jc      short sysvideo_29_5
```

```
11619 0000FE1D 890D[348E0100]      <1>     mov     [u.r0], ecx
11620 0000FE23 88D9                <1>     mov     cl, bl ; mov cl,7 (mov cl,3)
11621 0000FE25 89FE                <1>     mov     esi, edi ; VBE3SAVERESTOREBLOCK
11622                              <1>     ; cl = 3 or 7
11623 0000FE27 E8DF3EFFFF          <1>     call    biosfn_restore_video_state
11624 0000FE2C EBB6                <1>     jmp     sysvideo_29_5
11625                              <1>     ;jmp    sysret
11626                              <1> sysvideo_29_9:
11627                              <1>     ; Save
11628                              <1>     ;mov    edi, VBE3SAVERESTOREBLOCK
11629                              <1>
11630 0000FE2E 80FB06              <1>     cmp     bl, 6
11631 0000FE31 7409                <1>     je      short sysvideo_29_10
11632                              <1>     ; bl = 4
11633 0000FE33 BB6E000000          <1>     mov     ebx, 110
11634 0000FE38 B103                <1>     mov     cl, 3 ; ctrl, vbios data
11635 0000FE3A EB07                <1>     jmp     short sysvideo_29_11
11636                              <1> sysvideo_29_10:
11637                              <1>     ; bl = 6
11638 0000FE3C BB72030000          <1>     mov     ebx, 882
11639 0000FE41 B107                <1>     mov     cl, 7 ; ctrl, vbios data, dac
11640                              <1> sysvideo_29_11:
11641 0000FE43 E85B3DFFFF          <1>     call    biosfn_save_video_state
11642                              <1>
11643 0000FE48 89D9                <1>     mov     ecx, ebx ; transfer count
11644 0000FE4A 89F7                <1>     mov     edi, esi
11645 0000FE4C BE00760900          <1>     mov     esi, VBE3SAVERESTOREBLOCK
11646                              <1>
11647                              <1>     ;call   transfer_to_user_buffer
11648                              <1>     ;jc     short sysvideo_29_5
11649                              <1>     ;mov    [u.r0], ecx ; transfer count
11650                              <1>     ;;jmp   sysret
11651                              <1>     ;jmp    short sysvideo_29_5
11652                              <1>
11653 0000FE51 EB1A                <1>     jmp     short sysvideo_29_12
11654                              <1>
11655                              <1> sysvideo_30:
11656 0000FE53 80FF0F              <1>     cmp     bh, 15
11657 0000FE56 7722                <1>     ja      short sysvideo_31 ; invalid function
11658                              <1>
11659                              <1>     ; BH = 15
11660                              <1>     ; Copy VESA EDID to user's buffer
11661                              <1>
11662 0000FE58 803D[19420000]4F    <1>     cmp     byte [edid], 4Fh
11663 0000FE5F 7519                <1>     jne     short sysvideo_31 ; not ready !
11664                              <1>
11665                              <1>     ;and    ecx, ecx
11666                              <1>     ;jz     short sysvideo_31
11667                              <1>
11668                              <1>     ; ecx = user's buffer address
11669 0000FE61 89CF                <1>     mov     edi, ecx
11670 0000FE63 BE[6C9C0100]        <1>     mov     esi, edid_info
11671 0000FE68 B980000000          <1>     mov     ecx, 128 ; 128 bytes
11672                              <1> sysvideo_29_12:
11673 0000FE6D E8780C0000          <1>     call    transfer_to_user_buffer
11674 0000FE72 7206                <1>     jc      short sysvideo_31
11675                              <1>
11676 0000FE74 890D[348E0100]      <1>     mov     [u.r0], ecx ; EDID size, 128 bytes
11677                              <1> sysvideo_31:
11678 0000FE7A E9FFCBFFFF          <1>     jmp     sysret
11679                              <1>
11680                              <1> sysexec:
11681                              <1>     ; 07/07/2025 - TRDOS 386 v2.0.10
11682                              <1>     ; 21/08/2024 - TRDOS 386 v2.0.9
11683                              <1>     ; 23/07/2022 - TRDOS 386 v2.0.5
11684                              <1>     ; 06/02/2022 - Retro UNIX 386 v1.2
11685                              <1>     ; 18/11/2017
11686                              <1>     ; 14/11/2017
11687                              <1>     ; 13/11/2017
11688                              <1>     ; 24/10/2016 - 04/01/2017
11689                              <1>     ; 24/04/2016 - TRDOS 386 (TRDOS v2.0)
11690                              <1>     ; 23/06/2015 - 23/10/2015 (Retro UNIX 386 v1)
11691                              <1>     ; 03/06/2013 - 06/12/2013 (Retro UNIX 8086 v1)
11692                              <1>     ;
11693                              <1>     ; 'sysexec' initiates execution of a file whose path name if
11694                              <1>     ; pointed to by 'name' in the sysexec call.
11695                              <1>     ; 'sysexec' performs the following operations:
11696                              <1>     ;    1. obtains i-number of file to be executed via 'namei'.
11697                              <1>     ;    2. obtains i-node of file to be exceuted via 'iget'.
11698                              <1>     ;    3. sets trap vectors to system routines.
11699                              <1>     ;    4. loads arguments to be passed to executing file into
11700                              <1>     ;       highest locations of user's core
11701                              <1>     ;    5. puts pointers to arguments in locations immediately
11702                              <1>     ;       following arguments.
11703                              <1>     ;    6. saves number of arguments in next location.
11704                              <1>     ;    7. initializes user's stack area so that all registers
11705                              <1>     ;       will be zeroed and the PS is cleared and the PC set
11706                              <1>     ;       to core when 'sysret' restores registers
11707                              <1>     ;       and does an rti.
11708                              <1>     ;    8. initializes u.r0 and u.sp
11709                              <1>     ;    9. zeros user's core down to u.r0
11710                              <1>     ;   10. reads executable file from storage device into core
11711                              <1>     ;       starting at location 'core'.
11712                              <1>     ;   11. sets u.break to point to end of user's code with
11713                              <1>     ;       data area appended.
11714                              <1>     ;   12. calls 'sysret' which returns control at location
11715                              <1>     ;       'core' via 'rti' instruction.
11716                              <1>     ;
11717                              <1>     ; Calling sequence:
11718                              <1>     ;       sysexec; namep; argp
11719                              <1>     ; Arguments:
11720                              <1>     ;       namep - points to pathname of file to be executed
11721                              <1>     ;       argp  - address of table of argument pointers
11722                              <1>     ;       argp1... argpn - table of argument pointers
11723                              <1>     ;       argp1:<...0> ... argpn:<...0> - argument strings
11724                              <1>     ; Inputs: (arguments)
11725                              <1>     ; Outputs: -
11726                              <1>     ; ..........................................................
11727                              <1>     ;
11728                              <1>     ; Retro UNIX 386 v1 modification:
11729                              <1>     ;       User application runs in it's own virtual space
11730                              <1>     ;       which is izolated from kernel memory (and other
11731                              <1>     ;       memory pages) via 80386       paging in ring 3
11732                              <1>     ;       privilige mode. Virtual start address is always 0.
11733                              <1>     ;       User's core memory starts at linear address 400000h
11734                              <1>     ;       (the end of the 1st 4MB).
11735                              <1>     ;
11736                              <1>     ; Retro UNIX 8086 v1 modification:
11737                              <1>     ;       user/application segment and system/kernel segment
11738                              <1>     ;       are different and sysenter/sysret/sysrele routines
11739                              <1>     ;       are different (user's registers are saved to
11740                              <1>     ;       and then restored from system's stack.)
11741                              <1>     ;
11742                              <1>     ;       NOTE: Retro UNIX 8086 v1 'arg2' routine gets these
```

```
11743                                  <1>     ;            arguments which were in these registers;
11744                                  <1>     ;            but, it returns by putting the 1st argument
11745                                  <1>     ;            in 'u.namep' and the 2nd argument
11746                                  <1>     ;            on top of stack. (1st argument is offset of the
11747                                  <1>     ;            file/path name in the user's program segment.)
11748                                  <1>
11749                                  <1>     ;call  arg2
11750                                  <1>     ; * name - 'u.namep' points to address of file/path name
11751                                  <1>     ;          in the user's program segment ('u.segmnt')
11752                                  <1>     ;          with offset in BX register (as sysopen argument 1).
11753                                  <1>     ; * argp - sysexec argument 2 is in CX register
11754                                  <1>     ;          which is on top of stack.
11755                                  <1>     ;
11756                                  <1>             ; jsr r0,arg2 / arg0 in u.namep,arg1 on top of stack
11757                                  <1>
11758                                  <1>     ; 23/06/2015 (32 bit modifications)
11759                                  <1>
11760                                  <1>     ;; 13/11/2017
11761                                  <1>     ;;mov  [u.namep], ebx ; argument 1
11762                                  <1>        ; 18/10/2015
11763 0000FE7F 890D[348F0100]          <1>     mov     [argv], ecx  ; * ; argument 2
11764                                  <1>
11765                                  <1>     ; 13/11/2017
11766 0000FE85 89DE                    <1>     mov     esi, ebx
11767 0000FE87 E8181E0000              <1>     call    set_working_path_x
11768 0000FE8C 7319                    <1>     jnc     short sysexec_0
11769                                  <1>
11770                                  <1>     ;; 'bad command or file name'
11771                                  <1>     ;mov    eax, ERR_BAD_CMD_ARG ; 01h ; TRDOS 8086
11772                                  <1>
11773                                  <1>     ; 'file not found !' error
11774 0000FE8E B802000000              <1>     mov     eax, ERR_NOT_FOUND ; 02h ; TRDOS 8086
11775                                  <1> sysexec_not_found_err:
11776                                  <1> sysexec_access_error:
11777                                  <1> sysexec_ext_error:
11778 0000FE93 A3[348E0100]            <1>     mov     [u.r0], eax
11779 0000FE98 A3[A08E0100]            <1>     mov     [u.error], eax
11780 0000FE9D E8D71E0000              <1>     call    reset_working_path
11781 0000FEA2 E9B7CBFFFF              <1>     jmp     error
11782                                  <1>
11783                                  <1> sysexec_0:
11784                                  <1>     ; 13/11/2017
11785                                  <1>     ;mov    esi, FindFile_Name
11786 0000FEA7 66B80018                <1>        mov ax, 1800h ; Only files
11787 0000FEAB E8BA8AFFFF              <1>     call    find_first_file
11788 0000FEB0 72E1                    <1>     jc      short sysexec_not_found_err ; eax = 2
11789                                  <1>
11790                                  <1>     ; check_ file attributes
11791                                  <1>     ; (attribute bits = 00ADVSHR) ; 18h = Directory+Volume
11792                                  <1>     ; BL = Attributes byte
11793                                  <1>
11794 0000FEB2 F6C306                  <1>        test       bl, 6  ; system file or hidden file (S+H)
11795                                  <1>     ;jz     short sysexec_0ext
11796 0000FEB5 7417                    <1>     jz      short sysexec_1 ; yes
11797                                  <1>
11798                                  <1>     ; 13/11/2017
11799                                  <1>     ; /// TRDOS386 permission check for multiuser mode ///
11800                                  <1>     ; SYSTEM file or HIDDEN file !!
11801                                  <1>     ; (Only super user has permission to run this file.)
11802                                  <1>
11803                                  <1>     ; ([u.uid]=0 for super user or root in multiuser mode)
11804                                  <1>     ; ([u.uid]=0 for any users in singleuser mode)
11805 0000FEB7 803D[868E0100]00         <1>     cmp     byte [u.uid], 0 ; Super User ([u.uid]=0) ?
11806                                  <1>     ;jna    short sysexec_0ext
11807 0000FEBE 760E                    <1>     jna     short sysexec_1 ; yes
11808                                  <1>
11809                                  <1>     ; 'permission denied !' error
11810 0000FEC0 B80B000000              <1>        mov eax, ERR_FILE_ACCESS  ; 11 = ERR_PERM_DENIED
11811 0000FEC5 EBCC                    <1>        jmp short sysexec_access_error
11812                                  <1>
11813                                  <1> sysexec_not_exf:
11814                                  <1>     ; 'not executable file !' error
11815 0000FEC7 B816000000              <1>     mov     eax, ERR_NOT_EXECUTABLE
11816 0000FECC EBC5                    <1>     jmp     sysexec_ext_error
11817                                  <1>
11818                                  <1> ;sysexec_0ext:
11819                                  <1> sysexec_1:
11820                                  <1>     ; 18/11/2017
11821 0000FECE BE[24800100]            <1>     mov     esi, FindFile_Name
11822                                  <1>     ; 13/11/2017
11823                                  <1>     ; check program file name extension
11824                                  <1>     ; ('.PRG' for current TRDOS version)
11825 0000FED3 E8F8A3FFFF              <1>     call    check_prg_filename_ext
11826 0000FED8 72ED                    <1>     jc      short sysexec_not_exf
11827                                  <1>
11828                                  <1>     ; 18/11/2017
11829 0000FEDA 3C50                    <1>     cmp     al, 'P'
11830 0000FEDC 75E9                    <1>     jne     short sysexec_not_exf
11831                                  <1>
11832                                  <1>     ; '.PRG' extension is OK.
11833                                  <1>     ; Only '.PRG' files are valid program files
11834                                  <1>     ; for current TRDOS 386 version.
11835                                  <1>
11836 0000FEDE 8B15[50800100]          <1>     mov     edx, [FindFile_DirEntry+DirEntry_FileSize]
11837 0000FEE4 66A1[48800100]          <1>     mov     ax, [FindFile_DirEntry+DirEntry_FstClusHI]
11838 0000FEEA C1E010                  <1>     shl     eax, 16
11839 0000FEED 66A1[4E800100]          <1>     mov     ax, [FindFile_DirEntry+DirEntry_FstClusLO]
11840                                  <1>     ; EAX = First Cluster number
11841                                  <1>     ; EDX = File Size
11842                                  <1>
11843 0000FEF3 A3[3C8F0100]            <1>     mov     [ii], eax
11844 0000FEF8 8915[408F0100]          <1>     mov     [i.size], edx
11845                                  <1>
11846                                  <1> ;sysexec_1:
11847                                  <1>     ; 23/07/2022 - TRDOS 386 Kernel v2.0.5
11848                                  <1>     ; 06/02/2022 - Retro UNIX 386 v1.2
11849                                  <1>     ; 13/11/2017 - TRDOS 386 (TRDOS v2.0)
11850                                  <1>     ; 24/06/2015 - 23/10/2015 (Retro UNIX 386 v1)
11851                                  <1>        ; Moving arguments to the end of [u.upage]
11852                                  <1>     ; (by regarding page borders in user's memory space)
11853                                  <1>     ;
11854                                  <1>     ; 10/10/2015
11855                                  <1>     ; 21/07/2015
11856                                  <1>     ;mov    ebp, esp ; (**)
11857                                  <1>     ; 18/10/2015
11858                                  <1>     ;mov    edi, ebp
11859                                  <1>     ; 23/07/2022
11860 0000FEFE 89E7                    <1>     mov     edi, esp ; (**)
11861 0000FF00 B900010000              <1>     mov     ecx, MAX_ARG_LEN ; 256
11862                                  <1>     ;sub    edi, MAX_ARG_LEN ; 256
11863 0000FF05 29CF                    <1>     sub     edi, ecx
11864 0000FF07 89FC                    <1>     mov     esp, edi ; *!*
11865 0000FF09 31C0                    <1>     xor     eax, eax
11866 0000FF0B A3[608E0100]            <1>     mov     [u.nread], eax ; 0
```

```
11867                                   <1>     ; ([argc] must be cleared because previous 'sysexec'
11868                                   <1>     ; may leave it with any value after an error))
11869                                   <1>     ;mov    [argc], ax ; 0 ; 13/11/2017
11870                                   <1>     ; 23/07/2022
11871 0000FF10 A3[308F0100]             <1>     mov     [argc], eax ; 0
11872 0000FF15 49                       <1>     dec     ecx ; 256 - 1
11873 0000FF16 890D[5C8E0100]           <1>     mov     [u.count], ecx ; MAX_ARG_LEN - 1 ; 255
11874                                   <1>     ;mov    dword [u.count], MAX_ARG_LEN - 1 ; 255
11875                                   <1> sysexec_2:
11876 0000FF1C 8B35[348F0100]           <1>     mov     esi, [argv] ; 18/10/2015
11877 0000FF22 E86E020000               <1>     call    get_argp
11878                                   <1>     ;mov    ecx, 4
11879                                   <1>     ; 23/07/2022
11880 0000FF27 31C9                     <1>     xor     ecx, ecx
11881 0000FF29 B104                     <1>     mov     cl, 4
11882                                   <1> sysexec_3:
11883 0000FF2B 21C0                     <1>     and     eax, eax
11884 0000FF2D 7453                     <1>         jz  short sysexec_6 ; 23/07/2022
11885                                   <1>     ; 18/10/2015
11886 0000FF2F 010D[348F0100]           <1>     add     [argv], ecx ; 4
11887                                   <1>     ;;inc   word [argc]
11888                                   <1>     ; 23/07/2022
11889                                   <1>     ;inc    dword [argc]
11890 0000FF35 FE05[308F0100]           <1>     inc     byte [argc]
11891                                   <1>     ;
11892 0000FF3B A3[588E0100]             <1>     mov     [u.base], eax
11893                                   <1>     ; 23/10/2015
11894 0000FF40 66C705[988E0100]00-      <1>     mov     word [u.pcount], 0
11894 0000FF48 00                       <1>
11895                                   <1> sysexec_4:
11896 0000FF49 E83D0B0000               <1>     call    cpass ; get a character from user's core memory
11897 0000FF4E 750B                     <1>         jnz short sysexec_5
11898                                   <1>             ; (max. 255 chars + null)
11899                                   <1>     ; 18/10/2015
11900 0000FF50 28C0                     <1>     sub     al, al
11901 0000FF52 AA                       <1>     stosb
11902 0000FF53 FF05[608E0100]           <1>     inc     dword [u.nread]
11903                                   <1>     ; 23/07/2022
11904 0000FF59 EB27                     <1>     jmp     short sysexec_6 ; 24/04/2016
11905                                   <1> sysexec_5:
11906 0000FF5B AA                       <1>     stosb
11907 0000FF5C 20C0                     <1>     and     al, al
11908 0000FF5E 75E9                     <1>     jnz     short sysexec_4
11909                                   <1>     ;mov    ecx, 4
11910                                   <1>     ; 23/07/2022
11911 0000FF60 31C9                     <1>     xor     ecx, ecx
11912 0000FF62 B104                     <1>     mov     cl, 4
11913 0000FF64 390D[2C8F0100]           <1>     cmp     [ncount], ecx ; 4
11914 0000FF6A 72B0                     <1>     jb      short sysexec_2
11915 0000FF6C 8B35[288F0100]           <1>     mov     esi, [nbase]
11916 0000FF72 010D[288F0100]           <1>     add     [nbase], ecx ; 4
11917                                   <1>     ;sub    [ncount], cx
11918                                   <1>     ; 23/07/2022
11919 0000FF78 290D[2C8F0100]           <1>     sub     [ncount], ecx
11920 0000FF7E 8B06                     <1>     mov     eax, [esi]
11921 0000FF80 EBA9                     <1>     jmp     short sysexec_3
11922                                   <1>
11923                                   <1> sysexec_6:
11924                                   <1>     ; 23/07/2022 - TRDOS 386 Kernel v2.0.5
11925                                   <1>     ; 19/11/2017
11926                                   <1>     ; 18/11/2017
11927                                   <1>     ; 14/11/2017
11928                                   <1>     ; 13/11/2017
11929 0000FF82 8925[348F0100]           <1>     mov     [argv], esp ; *!* ; start address of argument list
11930                                   <1>
11931                                   <1>     ; 04/01/2017
11932                                   <1>     ; 24/10/2016
11933                                   <1>     ;;02/05/2016
11934                                   <1>     ; 23/04/2016 (TRDOS 386)
11935                                   <1>     ; 18/10/2015 ('sysexec_6')
11936                                   <1>     ; 23/06/2015
11937 0000FF88 A1[8C8E0100]             <1>     mov     eax, [u.pgdir] ; physical address of page directory
11938                                   <1>     ;cmp    eax, [k_page_dir] ; TRDOS MainProg ?
11939                                   <1>     ;je     short sysexec_7
11940                                   <1>     ; 19/11/2017
11941 0000FF8D 8B1D[908E0100]           <1>     mov     ebx, [u.ppgdir] ; phy addr of the parent's page dir
11942 0000FF93 E8BE59FFFF               <1>     call    deallocate_page_dir
11943                                   <1> sysexec_7:
11944 0000FF98 E8EE58FFFF               <1>     call    make_page_dir
11945                                   <1>     ;jc     panic  ; allocation error
11946                                   <1>     ;               ; after a deallocation would be nonsence !?
11947                                   <1>     ; 23/07/2022
11948 0000FF9D 7243                     <1>     jc      short sysexec_panic
11949                                   <1>
11950                                   <1>     ; 24/07/2015
11951                                   <1>     ; map kernel pages (1st 4MB) to PDE 0
11952                                   <1>     ;     of the user's page directory
11953                                   <1>     ;     (It is needed for interrupts!)
11954                                   <1>     ; 18/10/2015
11955 0000FF9F 8B15[80760100]           <1>     mov     edx, [k_page_dir] ; Kernel's page directory
11956 0000FFA5 8B02                     <1>     mov     eax, [edx] ; physical address of
11957                                   <1>                        ; kernel's first page table (1st 4 MB)
11958                                   <1>                        ; (PDE 0 of kernel's page directory)
11959 0000FFA7 8B15[8C8E0100]           <1>     mov     edx, [u.pgdir]
11960 0000FFAD 8902                     <1>     mov     [edx], eax ; PDE 0 (1st 4MB)
11961                                   <1>     ;
11962                                   <1>     ; 20/07/2015
11963 0000FFAF BB00004000               <1>     mov     ebx, CORE ; start address = 0 (virtual) + CORE
11964                                   <1>     ; 18/10/2015
11965 0000FFB4 BE[208F0100]             <1>     mov     esi, pcore ; physical start address
11966                                   <1> sysexec_8:
11967 0000FFB9 B907000000               <1>     mov     ecx, PDE_A_USER + PDE_A_WRITE + PDE_A_PRESENT
11968 0000FFBE E8E658FFFF               <1>     call    make_page_table
11969                                   <1>     ;jc     panic
11970                                   <1>     ; 23/07/2022
11971 0000FFC3 721D                     <1>     jc      short sysexec_panic
11972                                   <1>     ;
11973                                   <1>     ;mov    ecx, PTE_A_USER + PTE_A_WRITE + PTE_A_PRESENT
11974 0000FFC5 E8ED58FFFF               <1>     call    make_page ; make new page, clear and set the pte
11975                                   <1>     ;jc     panic
11976                                   <1>     ; 23/07/2022
11977 0000FFCA 7216                     <1>     jc      short sysexec_panic
11978                                   <1>     ;
11979 0000FFCC 8906                     <1>     mov     [esi], eax ; 24/06/2015
11980                                   <1>     ; ebx = virtual address (24/07/2015)
11981                                   <1>     ; 23/07/2022
11982                                   <1>     ;call   add_to_swap_queue
11983                                   <1>     ; 18/10/2015
11984 0000FFCE 81FE[248F0100]           <1>     cmp     esi, ecore ; user's stack (last) page ?
11985 0000FFD4 7411                     <1>     je      short sysexec_9 ; yes
11986 0000FFD6 BE[248F0100]             <1>     mov     esi, ecore  ; physical address of the last page
11987                                   <1>     ; 20/07/2015
11988 0000FFDB BB00F0FFFF               <1>     mov     ebx, (ECORE - PAGE_SIZE) + CORE
11989                                   <1>     ; ebx = virtual end address + segment base address - 4K
```

```
11990 0000FFE0 EBD7                <1>         jmp     short sysexec_8
11991                              <1> sysexec_panic:
11992                              <1>     ; 23/07/2022
11993 0000FFE2 E9226EFFFF          <1>     jmp     panic
11994                              <1>
11995                              <1> sysexec_9:
11996                              <1>     ; 23/07/2022 - TRDOS 386 Kernel v2.0.5
11997                              <1>     ; 19/11/2017
11998                              <1>     ; 24/04/2016 (TRDOS 386 = TRDOS v2.0)
11999                              <1>     ; 25/06/2015 - 26/08/2015 - 18/10/2015
12000                              <1>     ; move arguments from kernel stack to [ecore]
12001                              <1>     ; (argument list/line will be copied from kernel stack
12002                              <1>     ; frame to the last (stack) page of user's core memory)
12003                              <1>     ; 18/10/2015
12004 0000FFE7 8B3D[248F0100]      <1>     mov     edi, [ecore]
12005 0000FFED 81C700100000        <1>     add     edi, PAGE_SIZE
12006                              <1>     ; 19/11/2017
12007                              <1>     ;sub    edi, 4
12008                              <1>     ;mov    dword [edi], 0
12009                              <1>     ;mov    ebx, edi
12010                              <1>     ;
12011                              <1>     ;movzx  eax, word [argc]
12012                              <1>     ;or     eax, eax
12013                              <1>     ;jz     short sysexec_13 ; 19/11/2017
12014                              <1>     ;;jnz   short sysexec_10
12015                              <1>     ;;mov   ebx, edi
12016                              <1>     ;;sub   ebx, 4
12017                              <1>     ;;mov   [ebx], eax ; 0
12018                              <1>     ;;jmp   short sysexec_13
12019                              <1>     ; 23/07/2022
12020                              <1>     ; [argc] < 32
12021 0000FFF3 A1[308F0100]        <1>     mov     eax, [argc]
12022 0000FFF8 09C0                <1>     or      eax, eax
12023 0000FFFA 7509                <1>     jnz     short sysexec_10
12024 0000FFFC 89FB                <1>     mov     ebx, edi
12025 0000FFFE 83EB04              <1>     sub     ebx, 4
12026 00010001 8903                <1>     mov     [ebx], eax ; 0
12027 00010003 EB47                <1>     jmp     short sysexec_13
12028                              <1>
12029                              <1> sysexec_10:
12030 00010005 8B0D[608E0100]      <1>     mov     ecx, [u.nread]
12031                              <1>     ; 13/11/2017
12032                              <1>     ;;mov   esi, TextBuffer ; 'load_and_execute_file'
12033                              <1>     ;mov    esi, esp        ; 'sysexec'
12034 0001000B 8B35[348F0100]      <1>     mov     esi, [argv] ; 24/04/2016 (TRDOS 386  = TRDOS v2.0)
12035                              <1>     ; 23/07/2022
12036 00010011 29CF                <1>     sub     edi, ecx ; page end address - argument list length
12037                              <1>     ;sub    ebx, ecx ; 19/11/2017
12038                              <1>
12039                              <1>     ;;;;
12040                              <1>     ; 23/07/2022
12041                              <1>     ; (move edi -backward- to dword boundary)
12042                              <1>     ; ((this will prevent 'general protection fault' error
12043                              <1>     ;  as result of a lodsd or dword move instruction
12044                              <1>     ;  at the end of argument list))
12045 00010013 83EF03              <1>     sub     edi, 3
12046 00010016 83E7FC              <1>     and     edi, ~3 ; (*)
12047                              <1>     ;;;
12048                              <1>
12049 00010019 89C2                <1>     mov     edx, eax
12050 0001001B FEC2                <1>     inc     dl ; argument count + 1 for argc value
12051 0001001D C0E202              <1>     shl     dl, 2  ; 4 * (argument count + 1)
12052                              <1>     ; edx <= 128
12053 00010020 89FB                <1>     mov     ebx, edi
12054                              <1>     ;mov    edi, ebx ; 19//11/2017
12055                              <1>     ; 23/07/2022 (*) - edi is already dword aligned -
12056                              <1>     ;and    bl, 0FCh ; 32 bit (dword) alignment
12057 00010022 29D3                <1>     sub     ebx, edx
12058 00010024 89FA                <1>     mov     edx, edi
12059 00010026 F3A4                <1>     rep     movsb
12060 00010028 89D6                <1>     mov     esi, edx
12061 0001002A 89DF                <1>     mov     edi, ebx
12062 0001002C BA00F0BFFF          <1>     mov     edx, ECORE - PAGE_SIZE ; virtual addr. of the last page
12063 00010031 2B15[248F0100]      <1>     sub     edx, [ecore] ; difference (virtual - physical)
12064 00010037 AB                  <1>     stosd  ; eax = argument count
12065                              <1> sysexec_11:
12066 00010038 89F0                <1>     mov     eax, esi
12067 0001003A 01D0                <1>     add     eax, edx
12068 0001003C AB                  <1>     stosd  ; eax = virtual address
12069                              <1>     ; 23/07/2022
12070 0001003D FE0D[308F0100]      <1>     dec     byte [argc]
12071                              <1>     ;dec    dword [argc]
12072                              <1>     ;;dec   word [argc] ; 14/11/2017
12073 00010043 7407                <1>     jz      short sysexec_13
12074                              <1> sysexec_12:
12075 00010045 AC                  <1>     lodsb
12076 00010046 20C0                <1>     and     al, al
12077 00010048 75FB                <1>     jnz     short sysexec_12
12078 0001004A EBEC                <1>     jmp     short sysexec_11
12079                              <1> sysexec_13:
12080                              <1>     ; 24/10/2016
12081                              <1>     ; 24/04/2016 - TRDOS 386 (TRDOS v2.0)
12082                              <1>     ; 23/06/2015 - 19/10/2015 (Retro UNIX 386 v1, 'sysexec_13')
12083                              <1>     ;
12084                              <1>     ; moving arguments to [ecore] is OK here..
12085                              <1>     ;
12086                              <1>     ; ebx = beginning addres of argument list pointers
12087                              <1>             ;       in user's stack
12088 0001004C 2B1D[248F0100]      <1>     sub     ebx, [ecore]
12089 00010052 81C300F0BFFF        <1>     add     ebx, (ECORE - PAGE_SIZE)
12090                              <1>                     ; end of core - 4096 (last page)
12091                              <1>                     ; (virtual address)
12092 00010058 891D[348F0100]      <1>     mov     [argv], ebx
12093 0001005E 891D[648E0100]      <1>     mov     [u.break], ebx ; available user memory
12094                              <1>     ;
12095 00010064 29C0                <1>     sub     eax, eax
12096 00010066 C705[5C8E0100]2000- <1>     mov     dword [u.count], 32 ; Executable file header size
12096 0001006E 0000                <1>
12097 00010070 C705[488E0100]-     <1>     mov     dword [u.fofp], u.off
12097 00010076 [548E0100]          <1>
12098 0001007A A3[548E0100]        <1>     mov     [u.off], eax ; 0
12099 0001007F A3[588E0100]        <1>     mov     [u.base], eax ; 0, start of user's core (virtual)
12100                              <1>     ; 24/10/2016
12101 00010084 A0[42770100]        <1>     mov     al, [Current_Drv]
12102 00010089 A2[198E0100]        <1>     mov     [cdev], al
12103                              <1>     ;
12104 0001008E A1[3C8F0100]        <1>     mov     eax, [ii] ; Fist Cluster of the Program (PRG) file
12105                              <1>     ; EAX = First cluster of the executable file
12106 00010093 E8BD050000          <1>     call    readi
12107                              <1>
12108 00010098 8B0D[648E0100]      <1>     mov     ecx, [u.break] ; top of user's stack (physical addr.)
12109 0001009E 890D[5C8E0100]      <1>     mov     [u.count], ecx ; save for overrun check
12110                              <1>     ;
12111 000100A4 8B0D[608E0100]      <1>     mov     ecx, [u.nread]
```

```
12112 000100AA 890D[648E0100]      <1>     mov     [u.break], ecx ; virtual address (offset from start)
12113 000100B0 80F920              <1>     cmp     cl, 32
12114 000100B3 7540                <1>         jne     short sysexec_15
12115                              <1>     ;;
12116                              <1>     ; Retro UNIX 386 v1 (32 bit) executable file header format
12117 000100B5 8B35[208F0100]      <1>     mov     esi, [pcore] ; start address of user's core memory
12118                              <1>                          ; (phys. start addr. of the exec. file)
12119 000100BB AD                  <1>     lodsd
12120 000100BC 663DEB1E            <1>     cmp     ax, 1EEBh ; EBH, 1Eh -> jump to +32
12121 000100C0 7533                <1>     jne     short sysexec_15
12122 000100C2 AD                  <1>     lodsd
12123 000100C3 89C1                <1>     mov     ecx, eax ; text (code) section size
12124 000100C5 AD                  <1>     lodsd
12125 000100C6 01C1                <1>     add     ecx, eax ; + data section size (initialized data)
12126 000100C8 89CB                <1>     mov     ebx, ecx
12127 000100CA AD                  <1>     lodsd
12128 000100CB 01C3                <1>     add     ebx, eax ; + bss section size (for overrun checking)
12129 000100CD 3B1D[5C8E0100]      <1>     cmp     ebx, [u.count]
12130 000100D3 7711                <1>     ja      short sysexec_14  ; program overruns stack !
12131                              <1>     ;
12132                              <1>     ; add bss section size to [u.break]
12133 000100D5 0105[648E0100]      <1>     add     [u.break], eax
12134                              <1>     ;
12135 000100DB 83E920              <1>     sub     ecx, 32  ; header size (already loaded)
12136                              <1>     ;cmp    ecx, [u.count]
12137                              <1>     ;jnb    short sysexec_16
12138 000100DE 890D[5C8E0100]      <1>     mov     [u.count], ecx ; required read count
12139 000100E4 EB29                <1>     jmp     short sysexec_16
12140                              <1> sysexec_14:
12141                              <1>     ; insufficient (out of) memory
12142 000100E6 C705[A08E0100]0400- <1>     mov     dword [u.error], ERR_MINOR_IM ; 1
12142 000100EE 0000                <1>
12143 000100F0 E969C9FFFF          <1>     jmp     error
12144                              <1> sysexec_15:
12145 000100F5 8B15[408F0100]      <1>         mov edx, [i.size] ; file size
12146 000100FB 29CA                <1>     sub     edx, ecx ; file size - loaded bytes
12147 000100FD 7626                <1>     jna     short sysexec_17 ; no need to next read
12148 000100FF 01D1                <1>     add     ecx, edx ; [i.size]
12149 00010101 3B0D[5C8E0100]      <1>     cmp     ecx, [u.count] ; overrun check (!)
12150 00010107 77DD                <1>     ja      short sysexec_14
12151 00010109 8915[5C8E0100]      <1>     mov     [u.count], edx
12152                              <1> sysexec_16:
12153 0001010F A1[3C8F0100]        <1>     mov     eax, [ii] ; first cluster
12154 00010114 E83C050000          <1>     call    readi
12155 00010119 8B0D[608E0100]      <1>     mov     ecx, [u.nread]
12156 0001011F 010D[648E0100]      <1>     add     [u.break], ecx
12157                              <1> sysexec_17:
12158                              <1>     ; 07/07/2025
12159                              <1>     ;mov    eax, [ii] ; first cluster
12160                              <1>     ;call   iclose
12161 00010125 31C0                <1>     xor     eax, eax
12162                              <1>
12163                              <1>     ; 21/08/2024
12164                              <1>     ;inc    al
12165                              <1>     ;mov    [u.intr], ax ; 1 (interrupt/time-out is enabled)
12166                              <1>     ;mov    [u.quit], ax ; 1 ('crtl+brk' signal is enabled)
12167                              <1>     ; 23/07/2022
12168                              <1>     ;dec    al
12169                              <1>     ; 21/08/2024
12170 00010127 48                  <1>     dec     eax
12171 00010128 66A3[808E0100]      <1>     mov     [u.intr], ax ; -1 ; 0FFFFh ; enable CTRL+CRK
12172 0001012E 40                  <1>     inc     eax
12173 0001012F 66A3[828E0100]      <1>     mov     [u.quit], ax ; 0 ; reset CTRL+BRK flag
12174                              <1>
12175                              <1>     ;cmp    dword [u.ppgdir], 0  ; is the caller MainProg (kernel) ?
12176 00010135 3905[908E0100]      <1>     cmp     [u.ppgdir], eax ; 0 ; 23/07/2022
12177 0001013B 770C                <1>     ja      short sysexec_18 ; no, the caller is user process
12178                              <1>     ; If the caller is kernel (MainProg), 'sysexec' will come here
12179 0001013D 8B15[80760100]      <1>     mov     edx, [k_page_dir] ; kernel's page directory
12180 00010143 8915[908E0100]      <1>     mov     [u.ppgdir], edx ; next time 'sysexec' must not come here
12181                              <1> sysexec_18:
12182                              <1>     ; 02/05/2016
12183                              <1>     ; 24/04/2016 (TRDOS 386 = TRDOS v2.0)
12184                              <1>     ; 18/10/2015 (Retro UNIX 386 v1)
12185                              <1>     ; 05/08/2015
12186                              <1>     ; 29/07/2015
12187                              <1>
12188                              <1> ;   ; **** arguments list test start - 19/11/2017
12189                              <1> ;   mov     ebp, [argv]
12190                              <1> ;   sub     ebp, ECORE - 4096
12191                              <1> ;   add     ebp, [ecore]
12192                              <1> ;
12193                              <1> ;   mov     ebx, [ebp]
12194                              <1> ;   mov     [argc], bx
12195                              <1> ;   add     ebp, 4
12196                              <1> ;   mov     byte [ccolor], 1Fh
12197                              <1> ;_zx0:
12198                              <1> ;   cmp     word [argc], 0
12199                              <1> ;   jna     short _zx2
12200                              <1> ;_zx1:
12201                              <1> ;   push    ebp
12202                              <1> ;   mov     esi, [ebp]
12203                              <1> ;
12204                              <1> ;   sub     esi, ECORE - 4096
12205                              <1> ;   add     esi, [ecore]
12206                              <1> ;
12207                              <1> ;   call    print_cmsg
12208                              <1> ;
12209                              <1> ;   dec     word [argc]
12210                              <1> ;   jz      short _zx2
12211                              <1> ;
12212                              <1> ;   mov     al, '.'
12213                              <1> ;   mov     bl, 07h
12214                              <1> ;   mov     bh, [u.ttyn]
12215                              <1> ;   call    _write_tty
12216                              <1> ;
12217                              <1> ;   pop     ebp
12218                              <1> ;   add     ebp, 4
12219                              <1> ;   jmp     short _zx1
12220                              <1> ;_zx2:
12221                              <1> ;   pop     ebp
12222                              <1> ;   mov     byte [ccolor], 07h
12223                              <1> ;   mov     eax, 1
12224                              <1> ;   ; **** arguments list test stop
12225                              <1> ;   Test result is OK! (there is not a wrong thing) - 19/11/2017
12226                              <1>
12227 00010149 8B2D[348F0100]      <1>     mov     ebp, [argv] ; user's stack pointer must point to argument
12228                              <1>                         ; list pointers (argument count)
12229 0001014F FA                  <1>     cli
12230 00010150 8B25[1C760100]      <1>         mov     esp, [tss.esp0] ; ring 0 (kernel) stack pointer
12231                              <1>     ;mov    esp, [u.sp] ; Restore Kernel stack
12232                              <1>                         ; for this process
12233                              <1>     ;add    esp, 20 ; --> EIP, CS, EFLAGS, ESP, SS
12234                              <1>     ;;xor   eax, eax ; 0
```

```
12235                                  <1>     ; 23/07/2022
12236                                  <1>     ;dec    al ; eax = 0
12237                                  <1>     ; eax = 0
12238                                  <1>
12239                                  <1>     ;mov    edx, UDATA
12240                                  <1>     ; 18/11/2017
12241 00010156 6A23                    <1>     push    UDATA ; user's stack segment
12242                                  <1>     ;push   edx
12243 00010158 55                      <1>     push    ebp ; user's stack pointer
12244                                  <1>                 ; (points to number of arguments)
12245                                  <1>
12246                                  <1>     ; 04/01/2017
12247                                  <1>     ; MainProg comes here while [sysflg]= 0FFh
12248                                  <1>     ; (but sysexec comes here while [sysflg]= 0)
12249 00010159 C605[288E0100]00        <1>     mov     byte [sysflg], 0 ; 04/01/2017
12250                                  <1>                              ; (timer_int sysflg control)
12251 00010160 FB                      <1>     sti
12252 00010161 9C                      <1>     pushfd  ; EFLAGS
12253                                  <1>             ; Set IF for enabling interrupts in user mode
12254                                  <1>     ;or     dword [esp], 200h
12255                                  <1>     ;
12256                                  <1>     ;mov    bx, UCODE
12257                                  <1>     ;push   bx ; user's code segment
12258 00010162 6A1B                    <1>     push    UCODE
12259                                  <1>     ;push   0
12260 00010164 50                      <1>     push    eax ; EIP (=0) - start address -
12261 00010165 8925[2C8E0100]          <1>     mov     [u.sp], esp ; 29/07/2015
12262                                  <1>     ; 05/08/2015
12263                                  <1>     ; Remedy of a General Protection Fault during 'iretd' is here !
12264                                  <1>     ; ('push dx' would cause to general protection fault,
12265                                  <1>     ; after 'pop ds' etc.)
12266                                  <1>     ;
12267                                  <1>     ;; push dx ; ds (UDATA)
12268                                  <1>     ;; push dx ; es (UDATA)
12269                                  <1>     ;; push dx ; fs (UDATA)
12270                                  <1>     ;; push dx ; gs (UDATA)
12271                                  <1>     ;
12272                                  <1>     ; This is a trick to prevent general protection fault
12273                                  <1>     ; during 'iretd' intruction at the end of 'sysrele' (in u1.s):
12274 0001016B 66BA2300                <1>     mov     dx, UDATA ; 19/11/2017
12275 0001016F 8EC2                    <1>     mov     es, dx ; UDATA
12276 00010171 06                      <1>     push    es ; ds (UDATA)
12277 00010172 06                      <1>     push    es ; es (UDATA)
12278 00010173 06                      <1>     push    es ; fs (UDATA)
12279 00010174 06                      <1>     push    es ; gs (UDATA)
12280 00010175 66BA1000                <1>     mov     dx, KDATA
12281 00010179 8EC2                    <1>     mov     es, dx
12282                                  <1>     ;
12283                                  <1>     ;; pushad simulation
12284 0001017B 89E5                    <1>     mov     ebp, esp ; esp before pushad
12285 0001017D 50                      <1>     push    eax ; eax (0)
12286 0001017E 50                      <1>     push    eax ; ecx (0)
12287 0001017F 50                      <1>     push    eax ; edx (0)
12288 00010180 50                      <1>     push    eax ; ebx (0)
12289 00010181 55                      <1>     push    ebp ; esp before pushad
12290 00010182 50                      <1>     push    eax ; ebp (0)
12291 00010183 50                      <1>     push    eax ; esi (0)
12292 00010184 50                      <1>     push    eax ; edi (0)
12293                                  <1>     ;
12294 00010185 A3[348E0100]            <1>     mov     [u.r0], eax ; eax = 0
12295 0001018A 8925[308E0100]          <1>     mov     [u.usp], esp
12296                                  <1>
12297                                  <1>     ; 14/11/2017
12298 00010190 E9EBC8FFFF              <1>     jmp     sysret0
12299                                  <1>
12300                                  <1> get_argp:
12301                                  <1>     ; 08/08/2022
12302                                  <1>     ; 23/07/2022 - TRDOS 386 Kernel v2.0.5
12303                                  <1>     ; 11/12/2021 - Retro UNIX 386 v1.2
12304                                  <1>     ; 14/11/2017 - TRDOS 386 (TRDOS v2.0)
12305                                  <1>     ; 18/10/2015 (nbase, ncount)
12306                                  <1>     ; 21/07/2015
12307                                  <1>     ; 24/06/2015 (Retro UNIX 386 v1)
12308                                  <1>     ; Get (virtual) address of argument from user's core memory
12309                                  <1>     ;
12310                                  <1>     ; INPUT:
12311                                  <1>     ;       esi = virtual address of argument pointer
12312                                  <1>     ; OUTPUT:
12313                                  <1>     ;       eax = virtual address of argument
12314                                  <1>     ;
12315                                  <1>     ; Modified registers: EAX, EBX, ECX, EDX, ESI
12316                                  <1>     ;
12317 00010195 833D[908E0100]00        <1>     cmp     dword [u.ppgdir], 0 ; /etc/init ?
12318                                  <1>                                 ; (the caller is kernel)
12319                                  <1>     ;jna    short get_argpk
12320                                  <1>     ; 08/08/2022
12321 0001019C 7705                    <1>     ja      short get_argp5
12322 0001019E E985000000              <1>     jmp     get_argpk
12323                                  <1> get_argp5:
12324 000101A3 89F3                    <1>             mov     ebx, esi
12325 000101A5 E85A5AFFFF              <1>     call    get_physical_addr ; get physical address
12326 000101AA 7253                    <1>         jc      short get_argp_err ; 23/07/2022
12327 000101AC A3[288F0100]            <1>     mov     [nbase], eax ; physical address
12328                                  <1>     ;mov    [ncount], cx ; remain byte count in page (1-4096)
12329                                  <1>     ; 23/07/2022
12330 000101B1 890D[2C8F0100]          <1>     mov     [ncount], ecx
12331                                  <1>     ;mov    eax, 4 ; 21/07/2015
12332 000101B7 31C0                    <1>     xor     eax, eax
12333 000101B9 B004                    <1>     mov     al, 4
12334                                  <1>     ;cmp    cx, ax ; 4
12335                                  <1>     ; 23/07/2022
12336 000101BB 39C1                    <1>     cmp     ecx, eax ; 4
12337 000101BD 7354                    <1>     jnb     short get_argp2
12338 000101BF 89F3                    <1>     mov     ebx, esi
12339 000101C1 01CB                    <1>     add     ebx, ecx
12340 000101C3 E83C5AFFFF              <1>     call    get_physical_addr ; get physical address
12341 000101C8 7235                    <1>     jc      short get_argp_err
12342                                  <1>     ;push   esi
12343 000101CA 89C6                    <1>     mov     esi, eax
12344                                  <1>     ;xchg   cx, [ncount]
12345                                  <1>     ; 23/07/2022
12346 000101CC 870D[2C8F0100]          <1>     xchg    ecx, [ncount]
12347 000101D2 8735[288F0100]          <1>     xchg    esi, [nbase]
12348 000101D8 B504                    <1>     mov     ch, 4
12349 000101DA 28CD                    <1>     sub     ch, cl
12350                                  <1> get_argp0:
12351 000101DC AC                      <1>     lodsb
12352                                  <1>     ;push   ax
12353                                  <1>     ; 23/07/2022
12354 000101DD 50                      <1>     push    eax
12355 000101DE FEC9                    <1>     dec     cl
12356 000101E0 75FA                    <1>         jnz     short get_argp0
12357 000101E2 8B35[288F0100]          <1>     mov     esi, [nbase]
12358                                  <1>     ; 21/07/2015
```

```
12359 000101E8 0FB6C5              <1>     movzx  eax, ch
12360 000101EB 0105[288F0100]      <1>     add    [nbase], eax
12361                              <1>     ;sub   [ncount], ax
12362                              <1>     ; 23/07/2022
12363 000101F1 2905[2C8F0100]      <1>     sub    [ncount], eax
12364                              <1> get_argp1:
12365 000101F7 AC                  <1>     lodsb
12366 000101F8 FECD                <1>     dec    ch
12367 000101FA 7445                <1>         jz      short get_argp3
12368                              <1>         ;push      ax
12369                              <1>     ; 23/07/2022
12370 000101FC 50                  <1>     push   eax
12371 000101FD EBF8                <1>     jmp     short get_argp1
12372                              <1> get_argp_err:
12373 000101FF A3[A08E0100]        <1>     mov    [u.error], eax
12374                              <1>     ; 14/11/2017
12375 00010204 B801000000          <1>     mov    eax, ERR_BAD_CMD_ARG ; 01h ; TRDOS 8086
12376 00010209 A3[348E0100]        <1>     mov    [u.r0], eax
12377 0001020E E94BC8FFFF          <1>     jmp    error
12378                              <1> get_argp2:
12379                              <1>     ; 21/07/2015
12380                              <1>     ;mov   eax, 4
12381 00010213 8B15[288F0100]      <1>     mov    edx, [nbase] ; 18/10/2015
12382 00010219 0105[288F0100]      <1>     add    [nbase], eax
12383                              <1>     ;sub   [ncount], ax
12384                              <1>     ; 23/07/2022
12385 0001021F 2905[2C8F0100]      <1>     sub    [ncount], eax
12386                              <1>     ;
12387 00010225 8B02                <1>     mov    eax, [edx]
12388 00010227 C3                  <1>     retn
12389                              <1> get_argpk:
12390                              <1>     ; Argument is in kernel's memory space
12391 00010228 66C705[2C8F0100]00- <1>     mov    word [ncount], PAGE_SIZE ; 4096
12391 00010230 10                  <1>
12392 00010231 8935[288F0100]      <1>     mov    [nbase], esi
12393 00010237 8305[288F0100]04    <1>     add    dword [nbase], 4
12394 0001023E 8B06                <1>     mov    eax, [esi] ; virtual addr. = physical addr.
12395 00010240 C3                  <1>     retn
12396                              <1> get_argp3:
12397 00010241 B103                <1>     mov    cl, 3
12398                              <1> get_argp4:
12399 00010243 C1E008              <1>     shl    eax, 8
12400                              <1>     ;pop   dx
12401                              <1>     ; 23/07/2022
12402 00010246 5A                  <1>     pop    edx
12403 00010247 88D0                <1>     mov    al, dl
12404 00010249 E2F8                <1>         loop    get_argp4
12405                              <1>     ;pop   esi
12406 0001024B C3                  <1>     retn
12407                              <1>
12408                              <1>     ; 23/07/2022
12409                              <1> %if 0
12410                              <1>
12411                              <1> sysstat:
12412                              <1>     ; 13/01/2017 - TRDOS 386 (TRDOS v2.0)
12413                              <1>     ; temporary !
12414                              <1>     mov    eax, ERR_INV_FNUMBER ; 'invalid function number !'
12415                              <1>         mov     [u.error], eax
12416                              <1>         mov     [u.r0], eax
12417                              <1>     jmp    error
12418                              <1>
12419                              <1> sysfstat:
12420                              <1>     ; 13/01/2017 - TRDOS 386 (TRDOS v2.0)
12421                              <1>     ; temporary !
12422                              <1>     mov    eax, ERR_INV_FNUMBER ; 'invalid function number !'
12423                              <1>         mov     [u.error], eax
12424                              <1>         mov     [u.r0], eax
12425                              <1>     jmp    error
12426                              <1>
12427                              <1> %endif
12428                              <1>
12429                              <1> fclose:
12430                              <1>     ; 24/04/2025 - TRDOS 386 Kernel v2.0.10
12431                              <1>     ; 18/09/2024 - TRDOS 386 Kernel v2.0.9
12432                              <1>     ; 23/07/2022 - TRDOS 386 Kernel v2.0.5
12433                              <1>     ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
12434                              <1>     ;
12435                              <1>     ; 18/06/2015 (Retro UNIX 386 v1 - Beginning)
12436                              <1>     ;            (32 bit offset pointer modification)
12437                              <1>     ; 19/04/2013 - 12/01/2014 (Retro UNIX 8086 v1)
12438                              <1>     ;
12439                              <1>     ; Given the file descriptor (index to the u.fp list)
12440                              <1>     ; 'fclose' first gets the i-number of the file via 'getf'.
12441                              <1>     ; If i-node is active (i-number > 0) the entry in
12442                              <1>     ; u.fp list is cleared. If all the processes that opened
12443                              <1>     ; that file close it, then fsp etry is freed and the file
12444                              <1>     ; is closed. If not a return is taken.
12445                              <1>     ; If the file has been deleted while open, 'anyi' is called
12446                              <1>     ; to see anyone else has it open, i.e., see if it is appears
12447                              <1>     ; in another entry in the fsp table. Upon return from 'anyi'
12448                              <1>     ; a check is made to see if the file is special.
12449                              <1>     ;
12450                              <1>     ; INPUTS ->
12451                              <1>     ;    r1 - contains the file descriptor (value=0,1,2...)
12452                              <1>     ;    u.fp - list of entries in the fsp table
12453                              <1>     ;    fsp - table of entries (4 words/entry) of open files.
12454                              <1>     ; OUTPUTS ->
12455                              <1>     ;    r1 - contains the same file descriptor
12456                              <1>     ;    r2 - contains i-number
12457                              <1>     ;
12458                              <1>     ; ((AX = R1))
12459                              <1>     ; ((Modified registers: EDX, EBX, ECX, ESI, EDI, EBP))
12460                              <1>     ;
12461                              <1>     ; Retro UNIX 8086 v1 modification : CF = 1
12462                              <1>     ;              if i-number of the file is 0. (error)
12463                              <1>     ;
12464                              <1>     ; TRDOS 386 (06/10/2016)
12465                              <1>     ;
12466                              <1>     ; INPUT:
12467                              <1>     ;      EAX = File Handle (File Descriptor, File Index)
12468                              <1>     ;
12469                              <1>     ; OUTPUT:
12470                              <1>     ;      CF = 1 -> File not open !
12471                              <1>     ;      CF = 0 -> OK!
12472                              <1>     ;           EBX = File Number (System)
12473                              <1>     ;           [cdev] = Logical DOS Drive Number
12474                              <1>     ;           EAX = File Handle/Number (user)
12475                              <1>     ;
12476                              <1>     ; Modified Registers: EBX
12477                              <1>
12478 0001024C 50                  <1>     push   eax ; File handle
12479                              <1>
12480 0001024D E83D000000          <1>     call   getf
12481                              <1>     ;jc    device_close ; eax = device number
```

```
12482                                   <1>     ; 17/04/2021 (temporary)
12483 00010252 7302                     <1>     jnc     short _fclose_0
12484                                   <1>     ; 24/04/2025 (BugFix for 'sysexit')
12485 00010254 58                       <1>     pop     eax
12486                                   <1>     ;jmp    rw2 ; file not open !
12487 00010255 C3                       <1>     retn
12488                                   <1> _fclose_0:
12489 00010256 80BB[E4830100]01         <1>     cmp     byte [ebx+OF_MODE], 1 ; open mode ; 0 = empty entry
12490 0001025D 7227                     <1>     jb      short fclose_1       ; 1 = read, 2 = write
12491                                   <1>
12492                                   <1>     ; 18/09/2024 (EMPTY FILE BugFix)
12493                                   <1>     ;cmp    eax, 1 ; is the first cluster number > 0
12494                                   <1>     ;jb     short fclose_1 ; no, this is empty entry
12495                                   <1>
12496                                   <1> fclose_0:
12497 0001025F FE8B[24840100]           <1>     dec     byte [ebx+OF_OPENCOUNT] ; decrement the number of processes
12498                                   <1>                                     ; that have opened the file
12499 00010265 791F                     <1>     jns     short fclose_1 ; jump if not negative (jump if bit 7 is 0)
12500                                   <1>                    ; if all processes haven't closed the file, return
12501                                   <1>     ;
12502                                   <1>     ; eax ; First cluster
12503 00010267 31C0                     <1>     xor     eax, eax ; 0
12504 00010269 8883[E4830100]           <1>     mov     [ebx+OF_MODE], al ; 0 = empty entry
12505                                   <1>     ;mov    [ebx+OF_STATUS], al ; 0 = empty entry
12506                                   <1>     ;shl    bx, 2
12507                                   <1>     ; 23/07/2022
12508                                   <1>     ;shl    bl, 2
12509 0001026F C1E302                  <1>     shl     ebx, 2
12510 00010272 8983[44830100]           <1>     mov     [ebx+OF_FCLUSTER], eax ; 0
12511 00010278 8983[C4860100]           <1>     mov     [ebx+OF_CCLUSTER], eax ; 0
12512                                   <1>     ;mov    [ebx+OF_CCINDEX], eax ; 0
12513                                   <1>     ; 23/07/2022
12514                                   <1>     ;mov    [ebx+OF_OPENCOUNT], al ; 0
12515 0001027E A3[488E0100]            <1>     mov     [u.fofp], eax ; 0
12516                                   <1>     ;shr    bx, 2
12517                                   <1>     ; 23/07/2022
12518                                   <1>     ;shr    bl, 2
12519 00010283 C1EB02                  <1>     shr     ebx, 2
12520                                   <1> fclose_1: ; 1:
12521 00010286 58                       <1>     pop     eax ; File handle (File Descriptor, File Index)
12522 00010287 C680[3E8E0100]00         <1>     mov     byte [eax+u.fp], 0 ; clear that entry in the u.fp list
12523 0001028E C3                       <1>     retn
12524                                   <1>
12525                                   <1> getf:
12526                                   <1>     ; 03/09/2024 - TRDOS 386 v2.0.9
12527                                   <1>     ; 23/07/2022 - TRDOS 386 v2.0.5
12528                                   <1>     ; 17/04/2021 - TRDOS 386 v2.0.4
12529                                   <1>     ;      (temporary modifications)
12530                                   <1>     ; 12/10/2016
12531                                   <1>     ; 11/10/2016
12532                                   <1>     ; 08/10/2016
12533                                   <1>     ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
12534                                   <1>     ; / get the device number and the i-number of an open file
12535                                   <1>     ; 13/05/2015
12536                                   <1>     ; 11/05/2015 (Retro UNIX 386 v1 - Beginning)
12537                                   <1>     ; 19/04/2013 - 18/11/2013 (Retro UNIX 8086 v1)
12538                                   <1>     ;
12539 0001028F 89C3                     <1>     mov     ebx, eax
12540                                   <1> getf1:
12541 00010291 83FB0A                  <1>     cmp     ebx, 10
12542 00010294 730A                     <1>         jnb short getf2
12543 00010296 8A9B[3E8E0100]           <1>     mov     bl, [ebx+u.fp]
12544 0001029C 08DB                     <1>     or      bl, bl
12545 0001029E 7504                     <1>     jnz     short getf3
12546                                   <1> getf2:
12547                                   <1>     ; 'File not open !' error (ax=0)
12548 000102A0 29C0                     <1>     sub     eax, eax
12549                                   <1>     ; 03/09/2024
12550 000102A2 F9                       <1>     stc
12551 000102A3 C3                       <1>     retn
12552                                   <1> getf3:
12553                                   <1>     ; 23/07/2022
12554                                   <1>     ;test   bl, 80h
12555                                   <1>     ;jnz    short getf5 ; device
12556 000102A4 FECB                     <1>     dec     bl ; 0 based
12557 000102A6 8A83[C4830100]           <1>     mov     al, [ebx+OF_DRIVE]
12558 000102AC A2[198E0100]             <1>     mov     [cdev], al
12559 000102B1 C0E302                   <1>     shl     bl, 2 ; *4 (dword offset)
12560                                   <1>     ; 23/07/2022
12561                                   <1>     ;shl    ebx, 2
12562 000102B4 8B83[C4840100]           <1>     mov     eax, [ebx+OF_SIZE]
12563 000102BA A3[408F0100]             <1>     mov     [i.size], eax ; file size
12564 000102BF 8D83[44840100]           <1>     lea     eax, [ebx+OF_POINTER] ; 12/10/2016
12565 000102C5 A3[488E0100]             <1>     mov     [u.fofp], eax
12566 000102CA 8B83[44830100]           <1>     mov     eax, [ebx+OF_FCLUSTER]
12567 000102D0 C0EB02                   <1>     shr     bl, 2 ; /4 (byte offset)
12568                                   <1>     ; 23/07/2022
12569                                   <1>     ;shr    ebx, 2
12570                                   <1>     ; 17/04/2021
12571 000102D3 F8                       <1>     clc
12572                                   <1> getf4:
12573 000102D4 C3                       <1>     retn
12574                                   <1> ;getf5:
12575                                   <1>     ; 17/04/2021
12576                                   <1>     ; (following code is disabled as temporary)
12577                                   <1>     ;
12578                                   <1>     ;; get device number
12579                                   <1>     ;and    bl, 7Fh ; 1 to 7Fh
12580                                   <1>     ;dec    bl ; 0 based (0 to 7Eh)
12581                                   <1>     ;mov    al, [ebx+DEV_DRIVER]
12582                                   <1>     ;mov    ch, [ebx+DEV_ACCESS]
12583                                   <1>     ;mov    cl, [ebx+DEV_OPENMODE]
12584                                   <1>     ;and    ch, 0FEh ; reset bit 0 ; dev_close
12585                                   <1>     ;
12586                                   <1>     ; 23/07/2022
12587                                   <1>     ;stc ; cf = 1
12588                                   <1>     ;retn
12589                                   <1>
12590                                   <1> trans_addr_nmbp:
12591                                   <1>     ; 18/10/2015
12592                                   <1>     ; 12/10/2015
12593 000102D5 8B2D[508E0100]           <1>     mov     ebp, [u.namep]
12594                                   <1> trans_addr_nm:
12595                                   <1>     ; Convert virtual (pathname) address to physical address
12596                                   <1>     ; (Retro UNIX 386 v1 feature only !)
12597                                   <1>     ; 18/10/2015
12598                                   <1>     ; 12/10/2015 (u.pnbase & u.pncount has been removed from code)
12599                                   <1>     ; 02/07/2015
12600                                   <1>     ; 17/06/2015
12601                                   <1>     ; 16/06/2015
12602                                   <1>     ;
12603                                   <1>     ; INPUTS:
12604                                   <1>     ;       ebp = pathname address (virtual) ; [u.namep]
12605                                   <1>     ;       [u.pgdir] = user's page directory
```

```
12606                                   <1>     ; OUTPUT:
12607                                   <1>     ;       esi = physical address of the pathname
12608                                   <1>     ;       ecx = remain byte count in the page
12609                                   <1>     ;
12610                                   <1>     ; (Modified registers: EAX, EBX, ECX, EDX, ESI)
12611                                   <1>     ;
12612 000102DB 833D[908E0100]00        <1>         cmp     dword [u.ppgdir], 0  ; /etc/init ? (sysexec)
12613 000102E2 7618                    <1>     jna     short trans_addr_nmk ; the caller is os kernel;
12614                                   <1>                                  ; it is already physical address
12615 000102E4 50                      <1>     push    eax
12616 000102E5 89EB                    <1>     mov     ebx, ebp ; [u.namep] ; pathname address (virtual)
12617 000102E7 E81859FFFF              <1>             call    get_physical_addr ; get physical address
12618 000102EC 7204                    <1>     jc      short tr_addr_nm_err
12619                                   <1>     ; 18/10/2015
12620                                   <1>     ; eax = physical address
12621                                   <1>     ; cx = remain byte count in page (1-4096)
12622                                   <1>             ; 12/10/2015 (cx = [u.pncount])
12623 000102EE 89C6                    <1>     mov     esi, eax ; 12/10/2015 (esi=[u.pnbase])
12624 000102F0 58                      <1>     pop     eax
12625 000102F1 C3                      <1>     retn
12626                                   <1>
12627                                   <1> tr_addr_nm_err:
12628 000102F2 A3[A08E0100]            <1>     mov     [u.error], eax
12629                                   <1>     ;pop    eax
12630 000102F7 E962C7FFFF              <1>     jmp     error
12631                                   <1>
12632                                   <1> trans_addr_nmk:
12633                                   <1>     ; 12/10/2015
12634                                   <1>     ; 02/07/2015
12635 000102FC 8B35[508E0100]          <1>     mov     esi, [u.namep]  ; [u.pnbase]
12636 00010302 66B90010                <1>     mov     cx, PAGE_SIZE ; 4096 ; [u.pncount]
12637 00010306 C3                      <1>     retn
12638                                   <1>
12639                                   <1> sysbreak:
12640                                   <1>     ; 06/09/2024 - TRDOS 386 v2.0.9
12641                                   <1>     ; 23/07/2022 - TRDOS 386 v2.0.5
12642                                   <1>     ; 18/10/2015
12643                                   <1>     ; 07/10/2015
12644                                   <1>     ; 23/06/2015 (Retro UNIX 386 v1 - Beginning)
12645                                   <1>     ; 20/06/2013 - 24/03/2014 (Retro UNIX 8086 v1)
12646                                   <1>     ;
12647                                   <1>     ; 'sysbreak' sets the programs break points.
12648                                   <1>     ; It checks the current break point (u.break) to see if it is
12649                                   <1>     ; between "core" and the stack (sp). If it is, it is made an
12650                                   <1>     ; even address (if it was odd) and the area between u.break
12651                                   <1>     ; and the stack is cleared. The new breakpoint is then put
12652                                   <1>     ; in u.break and control is passed to 'sysret'.
12653                                   <1>     ;
12654                                   <1>     ; Calling sequence:
12655                                   <1>     ;       sysbreak; addr
12656                                   <1>     ; Arguments: -
12657                                   <1>     ;
12658                                   <1>     ; Inputs: u.break - current breakpoint
12659                                   <1>     ; Outputs: u.break - new breakpoint
12660                                   <1>     ;       area between old u.break and the stack (sp) is cleared.
12661                                   <1>     ;
12662                                   <1>     ; ................................................................
12663                                   <1>     ; 06/09/2024 - TRDOS 386 v2.0.9 - Major Modification -
12664                                   <1>     ;
12665                                   <1>     ;       This system call performs "malloc" function as in C.
12666                                   <1>     ; Input:
12667                                   <1>     ;    EBX = new u.break address (virtual, user's space)
12668                                   <1>     ; (Note: Default/Initial u.break address is the start of the BSS
12669                                   <1>     ;        section. Or it is file size for PRG -flat image- files.)
12670                                   <1>     ; If EBX = 0FFFFFFFFh ; -1
12671                                   <1>     ;    Return current u.break in EAX
12672                                   <1>     ; Output:
12673                                   <1>     ;    If EBX input is -1, EAX = current u.break address (virtual)
12674                                   <1>     ;       otherwise, EAX = new u.break address
12675                                   <1>     ; Area between the new u.break and the old u.break is cleared.
12676                                   <1>     ; (Note: If the new break address cross overs user's esp,
12677                                   <1>     ;        this area <new break - old break> is not cleared.)
12678                                   <1>     ; If CF=1 at return, it means Memory Allocation Error.
12679                                   <1>     ;             (or "Insufficient Memory" error)
12680                                   <1>     ;       EAX = 0
12681                                   <1>     ; ................................................................
12682                                   <1>     ;
12683                                   <1>     ; Retro UNIX 8086 v1 modification:
12684                                   <1>     ;       The user/application program puts breakpoint address
12685                                   <1>     ;        in BX register as 'sysbreak' system call argument.
12686                                   <1>     ;       (argument transfer method 1)
12687                                   <1>     ;
12688                                   <1>     ;  NOTE: Beginning of core is 0 in Retro UNIX 8086 v1 !
12689                                   <1>     ;       ((!'sysbreak' is not needed in Retro UNIX 8086 v1!))
12690                                   <1>     ;  NOTE:
12691                                   <1>     ;       'sysbreak' clears extended part (beyond of previous
12692                                   <1>     ;       'u.break' address) of user's memory for original unix's
12693                                   <1>     ;       'bss' compatibility with Retro UNIX 8086 v1 (19/11/2013)
12694                                   <1>
12695                                   <1>             ; mov u.break,r1 / move users break point to r1
12696                                   <1>             ; cmp r1,$core / is it the same or lower than core?
12697                                   <1>             ; blos 1f / yes, 1f
12698                                   <1>     ; 23/06/2015
12699 00010307 8B2D[648E0100]          <1>     mov     ebp, [u.break] ; virtual address (offset)
12700                                   <1>
12701                                   <1>     ; 06/09/2024
12702 0001030D 31C0                    <1>     xor     eax, eax
12703 0001030F A3[348E0100]            <1>     mov     [u.r0], eax ; 0 ; default ('memory allocation error')
12704 00010314 48                      <1>     dec     eax ; -1
12705                                   <1>
12706                                   <1>     ; 06/09/2024 - TRDOS 386 v2.0.9
12707                                   <1>     ; get u.break address (for 'malloc' in c compiler)
12708                                   <1>     ; for PRG files:
12709                                   <1>     ; default/initial u.break address is file size
12710 00010315 39C3                    <1>     cmp     ebx, eax ; -1 ; 0FFFFFFFFh
12711 00010317 7214                    <1>     jb      short sysbreak_@
12712                                   <1>
12713                                   <1> sysbreak_4:
12714 00010319 89EB                    <1>     mov     ebx, ebp
12715                                   <1>     ;(allocates a new page for user if it is not present)
12716 0001031B E8E458FFFF              <1>     call    get_physical_addr ; get physical address
12717 00010320 72D0                    <1>     jc      short tr_addr_nm_err
12718                                   <1>
12719 00010322 892D[348E0100]          <1>     mov     [u.r0], ebp  ; start of bss space (to be allocated)
12720 00010328 E951C7FFFF              <1>     jmp     sysret
12721                                   <1>
12722                                   <1> sysbreak_@:
12723                                   <1>     ;and    ebp, ebp
12724                                   <1>     ;jz     short sysbreak_3
12725                                   <1>     ; Retro UNIX 386 v1 NOTE: u.break points to virtual address !!!
12726                                   <1>     ; (Even break point address is not needed for Retro UNIX 386 v1)
12727 0001032D 8B15[2C8E0100]          <1>     mov     edx, [u.sp] ; kernel stack at the beginning of sys call
12728 00010333 83C20C                  <1>     add     edx, 12 ; EIP -4-> CS -4-> EFLAGS -4-> ESP (user)
12729                                   <1>     ; 07/10/2015
```

```
12730                                   <1>     ;mov    [u.break], ebx ; virtual address !!!
12731                                   <1>     ;
12732                                   <1>     ; 06/09/2024
12733                                   <1>     ; ebp = old/current u.break
12734                                   <1>
12735 00010336 3B1A                    <1>     cmp     ebx, [edx] ; compare new break point with
12736                                   <1>                        ; with top of user's stack (virtual!)
12737 00010338 732B                    <1>     jnb     short sysbreak_3
12738                                   <1>             ; cmp r1,sp / is it the same or higher
12739                                   <1>                     ; / than the stack?
12740                                   <1>             ; bhis 1f / yes, 1f
12741 0001033A 89DE                    <1>     mov     esi, ebx
12742 0001033C 29EE                    <1>     sub     esi, ebp ; new break point - old break point
12743                                   <1>     ;jna    short sysbreak_3
12744                                   <1>     ; 06/09/2024
12745 0001033E 7706                    <1>     ja      short sysbreak_1
12746 00010340 74D7                    <1>     jz      short sysbreak_4
12747 00010342 F7DE                    <1>     neg     esi ; convert to positive number (big-small)
12748 00010344 89DD                    <1>     mov     ebp, ebx ; move small number (address) to ebp
12749                                   <1>
12750                                   <1> sysbreak_1:
12751                                   <1>     ; 06/09/2024
12752 00010346 53                      <1>     push    ebx
12753 00010347 89EB                    <1>     mov     ebx, ebp
12754 00010349 E8B658FFFF              <1>     call    get_physical_addr ; get physical address
12755 0001034E 5B                      <1>     pop     ebx
12756 0001034F 72A1                    <1>     jc      short tr_addr_nm_err ; 23/07/2022
12757                                   <1>     ; 18/10/2015
12758 00010351 89C7                    <1>     mov     edi, eax
12759 00010353 29C0                    <1>     sub     eax, eax ; 0
12760                                   <1>              ; ECX = remain byte count in page (1-4096)
12761 00010355 39CE                    <1>     cmp     esi, ecx
12762 00010357 7302                    <1>     jnb     short sysbreak_2
12763 00010359 89F1                    <1>     mov     ecx, esi
12764                                   <1> sysbreak_2:
12765 0001035B 29CE                    <1>     sub     esi, ecx
12766 0001035D 01CD                    <1>     add     ebp, ecx
12767 0001035F F3AA                    <1>     rep     stosb
12768 00010361 09F6                    <1>     or      esi, esi
12769 00010363 75E1                    <1>     jnz     short sysbreak_1
12770                                   <1>             ; bit $1,r1 / is it an odd address
12771                                   <1>             ; beq 2f / no, its even
12772                                   <1>             ; clrb (r1)+ / yes, make it even
12773                                   <1>     ; 2: / clear area between the break point and the stack
12774                                   <1>             ; cmp r1,sp / is it higher or same than the stack
12775                                   <1>             ; bhis 1f / yes, quit
12776                                   <1>             ; clr (r1)+ / clear word
12777                                   <1>             ; br 2b / go back
12778                                   <1>     ;pop    ebx
12779                                   <1> sysbreak_3: ; 1:
12780                                   <1>     ; 06/09/2024
12781 00010365 891D[648E0100]          <1>     mov     [u.break], ebx ; virtual address !!!
12782                                   <1>             ; jsr r0,arg; u.break / put the "address"
12783                                   <1>                     ; / in u.break (set new break point)
12784                                   <1>             ; br sysret4 / br sysret
12785                                   <1>     ; 06/09/2024
12786 0001036B 891D[348E0100]          <1>     mov     [u.r0], ebx ; return new break point in eax
12787 00010371 E908C7FFFF              <1>     jmp     sysret
12788                                   <1>
12789                                   <1> sysseek: ; / moves read write pointer in an fsp entry
12790                                   <1>     ; 27/12/2025 - TRDOS 386 v2.0.10
12791                                   <1>     ; 06/11/2016 - TRDOS 386 (TRDOS v2.0)
12792                                   <1>     ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
12793                                   <1>     ; 07/07/2013 - 05/08/2013 (Retro UNIX 8086 v1)
12794                                   <1>     ;
12795                                   <1>     ; 'sysseek' changes the r/w pointer of (3rd word of in an
12796                                   <1>     ; fsp entry) of an open file whose file descriptor is in u.r0.
12797                                   <1>     ; The file descriptor refers to a file open for reading or
12798                                   <1>     ; writing. The read (or write) pointer is set as follows:
12799                                   <1>     ;       * if 'ptrname' is 0, the pointer is set to offset.
12800                                   <1>     ;       * if 'ptrname' is 1, the pointer is set to its
12801                                   <1>     ;         current location plus offset.
12802                                   <1>     ;       * if 'ptrname' is 2, the pointer is set to the
12803                                   <1>     ;         size of file plus offset.
12804                                   <1>     ; The error bit (e-bit) is set for an undefined descriptor.
12805                                   <1>     ;
12806                                   <1>     ; Calling sequence:
12807                                   <1>     ;       sysseek; offset; ptrname
12808                                   <1>     ; Arguments:
12809                                   <1>     ;       offset - number of bytes desired to move
12810                                   <1>     ;                the r/w pointer
12811                                   <1>     ;       ptrname - a switch indicated above
12812                                   <1>     ;
12813                                   <1>     ; Inputs: r0 - file descriptor
12814                                   <1>     ;       ; 27/12/2025
12815                                   <1>     ;       EBX = File descriptor (number) -user-
12816                                   <1>     ;       ECX = Offset (plus)
12817                                   <1>     ;       EDX = 0 -> set to offset in ECX
12818                                   <1>     ;           = 1 -> set to current offset + offset in ECX
12819                                   <1>     ;           = 2 -> set to file size + offset in ECX
12820                                   <1>     ; Outputs: -
12821                                   <1>     ;       EAX = (new) File Offset ; 27/12/2025
12822                                   <1>     ; ..........................................................
12823                                   <1>     ;
12824                                   <1>     ; Retro UNIX 8086 v1 modification:
12825                                   <1>     ;       'sysseek' system call has three arguments; so,
12826                                   <1>     ;       * 1st argument, file descriptor is in BX (BL) register
12827                                   <1>     ;       * 2nd argument, offset is in CX register
12828                                   <1>     ;       * 3rd argument, ptrname/switch is in DX (DL) register
12829                                   <1>
12830 00010376 E823000000              <1>     call    seektell
12831                                   <1>     ; EAX = Current R/W pointer of the file
12832                                   <1>     ; EBX = [u.fofp]
12833                                   <1>     ; [u.base] = offset (ECX input)
12834                                   <1>
12835 0001037B 0305[588E0100]          <1>     add     eax, [u.base]
12836 00010381 8903                    <1>     mov     [ebx], eax
12837                                   <1>     ; 27/12/2025
12838                                   <1> sysseek_@:
12839 00010383 A3[348E0100]            <1>     mov     [u.r0], eax ; return (new) file offset
12840 00010388 E9F1C6FFFF              <1>     jmp     sysret
12841                                   <1>
12842                                   <1> systell: ; / get the r/w pointer
12843                                   <1>     ; 27/12/2025 - TRDOS 386 v2.0.10
12844                                   <1>     ; 06/11/2016 - TRDOS 386 (TRDOS v2.0) - temporary !-
12845                                   <1>     ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
12846                                   <1>     ; 07/07/2013 - 05/08/2013 (Retro UNIX 8086 v1)
12847                                   <1>     ;
12848                                   <1>     ; Retro UNIX 8086 v1 modification:
12849                                   <1>     ; ! 'systell' does not work in original UNIX v1,
12850                                   <1>     ;           it returns with error !
12851                                   <1>     ; Inputs: r0 - file descriptor
12852                                   <1>     ;       ; 27/12/2025
12853                                   <1>     ;       EBX = File descriptor (number) -user-
```

```
                                <1>     ;       ECX = ignored
                                <1>     ;       EDX = 0 -> return current file offset
                                <1>     ;           = 1 -> return current file offset
                                <1>     ;           = 2 -> return file size
                                <1>     ;           > 2 -> return current file offset
                                <1>     ;
                                <1>     ; Outputs: r0 - file r/w pointer
                                <1>
                                <1>     ;xor    ecx, ecx ; 0
                                <1>     ;
                                <1>     ; 27/12/2025
0001038D 83FA02                 <1>     cmp     edx, 2
00010390 7405                   <1>     je      short systell_@
00010392 BA01000000             <1>     mov     edx, 1 ; 05/08/2013
                                <1> systell_@:
                                <1>     ;call   seektell
00010397 E808000000             <1>     call    seektell0 ; 05/08/2013
                                <1>     ;; 06/11/2016
                                <1>     ;;mov   eax, [ebx]
                                <1>     ; 27/12/2025
                                <1>     ;mov    [u.r0], eax
                                <1>     ;jmp    sysret
0001039C EBE5                   <1>     jmp     short sysseek_@
                                <1>
                                <1> ; Original unix v1 'systell' system call:
                                <1>             ; jsr r0,seektell
                                <1>             ; br error4
                                <1>
                                <1> seektell:
                                <1>     ; 24/04/2025 - TRDOS 386 v2.0.10
                                <1>     ; 06/11/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>     ; 03/01/2016
                                <1>     ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
                                <1>     ; 07/07/2013 - 05/08/2013 (Retro UNIX 8086 v1)
                                <1>     ;
                                <1>     ; 'seektell' puts the arguments from sysseek and systell
                                <1>     ; call in u.base and u.count. It then gets the i-number of
                                <1>     ; the file from the file descriptor in u.r0 and by calling
                                <1>     ; getf. The i-node is brought into core and then u.count
                                <1>     ; is checked to see it is a 0, 1, or 2.
                                <1>     ; If it is 0 - u.count stays the same
                                <1>     ;          1 - u.count = offset (u.fofp)
                                <1>     ;         2 - u.count = i.size (size of file)
                                <1>     ;
                                <1>     ; !! Retro UNIX 8086 v1 modification:
                                <1>     ;       Argument 1, file descriptor is in BX;
                                <1>     ;       Argument 2, offset is in CX;
                                <1>     ;       Argument 3, ptrname/switch is in DX register.
                                <1>     ;
                                <1>     ; ((Return -> eax = base for offset (position = base+offset))
                                <1>     ;
0001039E 890D[588E0100]         <1>     mov     [u.base], ecx ; offset
                                <1> seektell0:
000103A4 8915[5C8E0100]         <1>     mov     [u.count], edx
                                <1>     ; EBX = file descriptor (file number)
000103AA E8E2FEFFFF             <1>     call    getf1
                                <1>     ; EAX = First cluster of the file
                                <1>     ; EBX = File number (Open file number)
                                <1>     ; [u.fofp] = Pointer to File pointer
                                <1>     ; [i.size] = File size
                                <1>
000103AF 09C0                   <1>     or      eax, eax
000103B1 7505                   <1>     jnz     short seektell1
                                <1>
                                <1>     ; 24/04/2025 - TRDOS 386 v2.0.10
                                <1>     ;mov    eax, ERR_FILE_NOT_OPEN
                                <1>     ;mov    [u.r0], eax
                                <1>     ;mov    dword [u.error], eax ; 'file not open !'
                                <1>     ;jmp    error
                                <1>     ; 24/04/2025
000103B3 E9E6CDFFFF             <1>     jmp     seektell_err ; sysclose_err
                                <1>
                                <1> seektell1:
000103B8 8B1D[488E0100]         <1>         mov     ebx, [u.fofp]
000103BE 803D[5C8E0100]01       <1>     cmp     byte [u.count], 1
000103C5 7705                   <1>     ja      short seektell2
000103C7 7409                   <1>     je      short seektell3
000103C9 31C0                   <1>     xor     eax, eax
000103CB C3                     <1>     retn
                                <1>
                                <1> seektell2:
000103CC A1[408F0100]           <1>         mov        eax, [i.size]
000103D1 C3                     <1>     retn
                                <1>
                                <1> seektell3:
000103D2 8B03                   <1>     mov     eax, [ebx]
000103D4 C3                     <1>     retn
                                <1>
                                <1> sysintr: ; / set interrupt handling
                                <1>     ; 21/08/2024 - TRDOS 386 v2.0.9 modification
                                <1>     ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
                                <1>     ; 07/07/2013 (Retro UNIX 8086 v1)
                                <1>     ;
                                <1>     ; 'sysintr' sets the interrupt handling value. It puts
                                <1>     ; argument of its call in u.intr then branches into 'sysquit'
                                <1>     ; routine. u.tty is checked if to see if a control tty exists.
                                <1>     ; If one does the interrupt character in the tty buffer is
                                <1>     ; cleared and 'sysret'is called. If one does not exits
                                <1>     ; 'sysret' is just called.
                                <1>     ;
                                <1>     ; Calling sequence:
                                <1>     ;       sysintr; arg
                                <1>     ; Argument:
                                <1>     ;       arg - if 0, interrupts (ASCII DELETE) are ignored.
                                <1>     ;           - if 1, interrupts cause their normal result
                                <1>     ;               i.e force an exit.
                                <1>     ;           - if arg is a location within the program,
                                <1>     ;             control is passed to that location when
                                <1>     ;             an interrupt occurs.
                                <1>     ; Inputs: -
                                <1>     ; Outputs: -
                                <1>     ; ..........................................................
                                <1>     ;
                                <1>     ; Retro UNIX 8086 v1 modification:
                                <1>     ;       'sysintr' system call sets u.intr to value of BX
                                <1>     ;       then branches into sysquit.
                                <1>
                                <1>     ; 21/08/2024 - TRDOS 386
                                <1>     ; enable/disable terminate (CTRL+BREAK) interrupt
                                <1>     ;
                                <1>     ; INPUT:
                                <1>     ;       bx = 0 -> disable CTRL+BREAK
                                <1>     ;                 -u.intr will be set to 0, u.quit will be ignored-
                                <1>     ;       bx > 0 -> enable CTRL+BREAK (also default at start)
```

```
12978                                  <1>     ;             -u.intr will be set to 0FFFFh, u.quit will be used-
12979                                  <1>     ;       NOTE: u.quit is flag for CTRL+BREAK key press status
12980                                  <1>     ;             -1 = pressed -termination request-, 0 = not pressed
12981                                  <1>     ; OUTPUT:
12982                                  <1>     ;       none
12983                                  <1>
12984                                  <1>     ;;;
12985                                  <1>     ; 21/08/2024
12986 000103D5 F7C3FFFF0000           <1>     test    ebx, 0FFFFh
12987 000103DB 7403                   <1>     jz      short sysintr_@ ; 0
12988                                  <1>     ;mov    bx, 0FFFFh
12989 000103DD 31DB                   <1>     xor     ebx, ebx
12990 000103DF 4B                     <1>     dec     ebx ; -1
12991                                  <1> sysintr_@:
12992                                  <1>     ;;;
12993 000103E0 66891D[808E0100]       <1>     mov     [u.intr], bx
12994                                  <1>             ; jsr r0,arg; u.intr / put the argument in u.intr
12995                                  <1>             ; br 1f / go into quit routine
12996 000103E7 E992C6FFFF             <1>     jmp     sysret
12997                                  <1>
12998                                  <1> sysquit:
12999                                  <1>     ; 21/08/2024 - TRDOS 386 v2.0.9 modification
13000                                  <1>     ;       get/reset QUIT (u.quit) status
13001                                  <1>     ;
13002                                  <1>     ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
13003                                  <1>     ; 07/07/2013 (Retro UNIX 8086 v1)
13004                                  <1>     ;
13005                                  <1>     ; 'sysquit' turns off the quit signal. it puts the argument of
13006                                  <1>     ; the call in u.quit. u.tty is checked if to see if a control
13007                                  <1>     ; tty exists. If one does the interrupt character in the tty
13008                                  <1>     ; buffer is cleared and 'sysret'is called. If one does not exits
13009                                  <1>     ; 'sysret' is just called.
13010                                  <1>     ;
13011                                  <1>     ; Calling sequence:
13012                                  <1>     ;       sysquit; arg
13013                                  <1>     ; Argument:
13014                                  <1>     ;       arg - if 0, this call diables quit signals from the
13015                                  <1>     ;               typewriter (ASCII FS)
13016                                  <1>     ;           - if 1, quits are re-enabled and cause execution to
13017                                  <1>     ;               cease and a core image to be produced.
13018                                  <1>     ;                i.e force an exit.
13019                                  <1>     ;           - if arg is an addres in the program,
13020                                  <1>     ;               a quit causes control to sent to that
13021                                  <1>     ;               location.
13022                                  <1>     ; Inputs: -
13023                                  <1>     ; Outputs: -
13024                                  <1>     ; ..........................................................
13025                                  <1>     ;
13026                                  <1>     ; Retro UNIX 8086 v1 modification:
13027                                  <1>     ;       'sysquit' system call sets u.quit to value of BX
13028                                  <1>     ;      then branches into 'sysret'.
13029                                  <1>
13030                                  <1>     ; 21/08/2024
13031                                  <1>     ;mov    [u.quit], bx
13032                                  <1>     ;jmp    sysret
13033                                  <1>             ; jsr r0,arg; u.quit / put argument in u.quit
13034                                  <1>     ;1:
13035                                  <1>             ; mov u.ttyp,r1 / move pointer to control tty buffer
13036                                  <1>                             ; / to r1
13037                                  <1>             ; beq sysret4 / return to user
13038                                  <1>             ; clrb 6(r1) / clear the interrupt character
13039                                  <1>                      ; / in the tty buffer
13040                                  <1>             ; br sysret4 / return to user
13041                                  <1>
13042                                  <1>     ; 21/08/2024 - TRDOS 386
13043                                  <1>     ; get/reset QUIT (u.quit) status
13044                                  <1>     ;
13045                                  <1>     ; INPUT:
13046                                  <1>     ;       none
13047                                  <1>     ; OUTPUT:
13048                                  <1>     ;       eax = 0 -> no CTRL+BREAK
13049                                  <1>     ;       eax = -1 -> CTRL+BREAK status (but not applied)
13050                                  <1>     ;                -u.intr may be 0-
13051                                  <1>     ;       NOTE: u.quit will be reset to 0
13052                                  <1>
13053 000103EC 31C0                   <1>     xor     eax, eax ; 0
13054 000103EE 663905[828E0100]       <1>     cmp     [u.quit], ax
13055 000103F5 7407                   <1>     jz      short sysquit_@ ; already 0
13056 000103F7 66A3[828E0100]         <1>     mov     [u.quit], ax ; 0 ; reset
13057 000103FD 48                     <1>     dec     eax ; -1
13058                                  <1> sysquit_@:
13059 000103FE A3[348E0100]           <1>     mov     [u.r0], eax
13060 00010403 E976C6FFFF             <1>     jmp     sysret
13061                                  <1>
13062                                  <1> %if 0
13063                                  <1>
13064                                  <1> anyi:
13065                                  <1>     ; 23/07/2022
13066                                  <1>     ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
13067                                  <1>     ; Major Modification!
13068                                  <1>     ; TRDOS 386 does not permit to delete a file while it is open
13069                                  <1>     ; The role of 'anyi' procedure has beeen changed to ensure that.
13070                                  <1>     ;
13071                                  <1>     ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
13072                                  <1>     ; 25/04/2013 (Retro UNIX 8086 v1)
13073                                  <1>     ;
13074                                  <1>     ; 'anyi' is called if a file deleted while open.
13075                                  <1>     ; "anyi" checks to see if someone else has opened this file.
13076                                  <1>     ;
13077                                  <1>     ; INPUTS ->
13078                                  <1>     ;    r1 - contains an i-number
13079                                  <1>     ;    fsp - start of table containing open files
13080                                  <1>     ;
13081                                  <1>     ; OUTPUTS ->
13082                                  <1>     ;    "deleted" flag set in fsp entry of another occurrence of
13083                                  <1>     ;          this file and r2 points 1st word of this fsp entry.
13084                                  <1>     ;    if file not found - bit in i-node map is cleared
13085                                  <1>     ;                        (i-node is freed)
13086                                  <1>     ;                all blocks related to i-node are freed
13087                                  <1>     ;               all flags in i-node are cleared
13088                                  <1>     ; ((AX = R1)) input
13089                                  <1>     ;
13090                                  <1>     ;    (Retro UNIX Prototype : 02/12/2012, UNIXCOPY.ASM)
13091                                  <1>       ;    ((Modified registers: EDX, ECX, EBX, ESI, EDI, EBP))
13092                                  <1>     ;
13093                                  <1>     ; / r1 contains an i-number
13094                                  <1>
13095                                  <1>     ; TRDOS 386 (06/10/2016)
13096                                  <1>     ;
13097                                  <1>     ; INPUT:
13098                                  <1>     ;      EAX = First Cluster
13099                                  <1>     ;       DL = Logical DOS Drive Number
13100                                  <1>     ;
13101                                  <1>     ; OUTPUT:
```

```
13102                                 <1>     ;       CF = 1 -> EBX = File Handle/Number/Index
13103                                 <1>     ;       CF = 0 -> EBX = 0
13104                                 <1>     ;
13105                                 <1>     ; Modified Registers: EBX
13106                                 <1>
13107                                 <1>     xor     ebx, ebx
13108                                 <1> anyi_0:
13109                                 <1>     cmp     byte [ebx+OF_MODE], 0 ; 0 = empty entry
13110                                 <1>     ja      short anyi_2 ; 1 (r), 2 (w) or 3 (r&w)
13111                                 <1> anyi_1:
13112                                 <1>     inc     bl
13113                                 <1>     cmp     bl, OPENFILES ; max. count of open files
13114                                 <1>     jb      short anyi_0
13115                                 <1>     xor     eax, eax
13116                                 <1>     retn
13117                                 <1> anyi_2:
13118                                 <1>     cmp     dl, [ebx+OF_DRIVE]
13119                                 <1>     jne     short anyi_1
13120                                 <1>     ;shl    bx, 2 ; *4 (dword offset)
13121                                 <1>     shl     ebx, 2 ; 23/07/2022
13122                                 <1>     cmp     eax, [ebx+OF_FCLUSTER]
13123                                 <1>     je      short anyi_3
13124                                 <1>     ;shr    bx, 2 ; /4 (byte offset)
13125                                 <1>     shr     ebx, 2 ; 23/07/2022
13126                                 <1>     jmp     short anyi_1
13127                                 <1> anyi_3:
13128                                 <1>     ;shr    bx, 2 ; /4 (bytes offset) (index)
13129                                 <1>     shr     ebx, 2 ; 23/07/2022
13130                                 <1>     stc
13131                                 <1>     retn
13132                                 <1>
13133                                 <1> %endif
13134                                 <1>
13135                                 <1> ; Retro UNIX 386 v1 Kernel (v0.2) - SYS9.INC
13136                                 <1> ; Last Modification: 09/12/2015
13137                                 <1>
13138                                 <1> syssleep:
13139                                 <1>     ; 28/08/2024 - TRDOS 386 v2.0.9
13140                                 <1>     ; 24/07/2022 - TRDOS 386 v2.0.5
13141                                 <1>     ; 29/06/2015 - (Retro UNIX 386 v1)
13142                                 <1>     ; 11/06/2014 - (Retro UNIX 8086 v1)
13143                                 <1>     ;
13144                                 <1>     ; Retro UNIX 8086 v1 feature only
13145                                 <1>     ; (INPUT -> none)
13146                                 <1>
13147                                 <1>     ; Temporary - 24/07/2022
13148 00010408 891D[348E0100]        <1>     mov     [u.r0], ebx
13149                                 <1>     ;
13150 0001040E 0FB61D[858E0100]      <1>     movzx   ebx, byte [u.uno] ; process number
13151 00010415 8AA3[478D0100]        <1>     mov     ah, [ebx+p.ttyc-1] ; current/console tty
13152 0001041B E892150000            <1>     call    sleep
13153                                 <1>
13154                                 <1>     ; 23/08/2024 - restore [u.usp]
13155                                 <1>     ; swap changes [u.usp] to esp which points to
13156                                 <1>     ; return of sleep
13157 00010420 8925[308E0100]        <1>     mov     [u.usp], esp
13158                                 <1>             ; points to user's regs on top ofn system stack
13159                                 <1>
13160                                 <1>     ; 24/07/2022
13161 00010426 FF05[348E0100]        <1>     inc     dword [u.r0] ; Temporary !
13162 0001042C E94DC6FFFF            <1>     jmp     sysret
13163                                 <1>
13164                                 <1> _vp_clr:
13165                                 <1>     ; Reset/Clear Video Page
13166                                 <1>     ;
13167                                 <1>     ; 24/07/2022 - TRDOS 386 v2.0.5
13168                                 <1>     ; 30/06/2015 - (Retro UNIX 386 v1)
13169                                 <1>     ; 21/05/2013 - 30/10/2013(Retro UNIX 8086 v1) (U0.ASM)
13170                                 <1>     ;
13171                                 <1>     ; Retro UNIX 8086 v1 feature only !
13172                                 <1>     ;
13173                                 <1>     ; INPUTS ->
13174                                 <1>     ;   BH = video page number
13175                                 <1>     ;
13176                                 <1>     ; OUTPUT ->
13177                                 <1>     ;   none
13178                                 <1>     ; ((Modified registers: EAX, BH, ECX, EDX, ESI, EDI))
13179                                 <1>     ;
13180                                 <1>     ; 04/12/2013
13181 00010431 28C0                  <1>     sub     al, al
13182                                 <1>     ; al = 0 (clear video page)
13183                                 <1>     ; bh = video page ; 13/05/2016
13184 00010433 B407                  <1>     mov     ah, 07h
13185                                 <1>     ; ah = 7 (attribute/color)
13186                                 <1>     ;xor    cx, cx ; 0, left upper column (cl) & row (cl)
13187                                 <1>     ;mov    dx, 184Fh ; right lower column & row (dl=24, dh=79)
13188                                 <1>     ; 24/07/2022
13189 00010435 31C9                  <1>     xor     ecx, ecx
13190 00010437 BA4F180000            <1>     mov     edx, 184Fh
13191 0001043C E8EC1BFFFF            <1>     call    _scroll_up
13192                                 <1>     ; bh = video page
13193                                 <1>     ;xor    dx, dx ; 0 (cursor position)
13194                                 <1>     ; 24/07/2022
13195 00010441 31D2                  <1>     xor     edx, edx
13196 00010443 E92B1FFFFF            <1>     jmp     _set_cpos
13197                                 <1>
13198                                 <1> sysmsg:
13199                                 <1>     ; 28/08/2024 - TRDOS 386 v2.0.9
13200                                 <1>     ; 23/07/2022 - TRDOS 386 v2.0.5
13201                                 <1>     ; 07/12/2020
13202                                 <1>     ; 05/12/2020
13203                                 <1>     ; 13/05/2016
13204                                 <1>     ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
13205                                 <1>     ; 01/07/2015 - 11/11/2015 (Retro UNIX 386 v1)
13206                                 <1>     ; Print user-application message on user's console tty
13207                                 <1>     ;
13208                                 <1>     ; Input -> EBX = Message address
13209                                 <1>     ;          ECX = Message length (max. 255)
13210                                 <1>     ;          DL = Color (IBM PC Rombios color attributes)
13211                                 <1>     ;
13212 00010448 81F9FF000000          <1>     cmp     ecx, MAX_MSG_LEN ; 255
13213                                 <1>     ;ja     sysret ; nothing to do with big message size
13214                                 <1>     ; 23/07/2022
13215 0001044E 7605                  <1>     jna     short sysmsg8
13216                                 <1> sysmsg7:
13217 00010450 E929C6FFFF            <1>     jmp     sysret
13218                                 <1> sysmsg8:    ; 23/07/2022
13219 00010455 08C9                  <1>     or      cl, cl
13220                                 <1>     ;jz     sysret
13221                                 <1>     ; 23/07/2022
13222 00010457 74F7                  <1>     jz      short sysmsg7
13223 00010459 20D2                  <1>     and     dl, dl
13224 0001045B 7502                  <1>     jnz     short sysmsg0
13225 0001045D B207                  <1>     mov     dl, 07h ; default color
```

```
13226                                   <1>             ; (black background, light gray character)
13227                                   <1> sysmsg0:
13228 0001045F 891D[588E0100]           <1>     mov     [u.base], ebx
13229 00010465 8815[AF760100]           <1>     mov     [ccolor], dl ; color attributes
13230 0001046B 89E5                     <1>     mov     ebp, esp ; save stack pointer
13231 0001046D 31DB                     <1>     xor     ebx, ebx ; 0
13232 0001046F 891D[608E0100]           <1>     mov     [u.nread], ebx ; 0
13233                                   <1>     ;
13234 00010475 381D[9A8E0100]           <1>     cmp     [u.kcall], bl ; 0
13235 0001047B 7770                     <1>     ja      short sysmsgk ; Temporary (01/07/2015)
13236                                   <1>     ;
13237 0001047D 890D[5C8E0100]           <1>     mov     [u.count], ecx
13238                                   <1>     ;inc    ecx ; + 00h ; ASCIIZ
13239                                   <1>     ;
13240                                   <1>     ; 07/12/2020
13241                                   <1>     ;add    ecx, 3
13242 00010483 6683C103                 <1>     add     cx, 3
13243 00010487 80E1FC                   <1>     and     cl, ~3  ; not 3
13244                                   <1>     ;
13245                                   <1>     ;;;
13246                                   <1>     ; 20/08/2024
13247                                   <1>     ; for safety, against a possibility which if ecx+1
13248                                   <1>     ; chars (without a 0 in ecx chars) overs stack frame
13249                                   <1>     ; (for example 255 chars without 0, safe stack frame
13250                                   <1>     ; size is 260.)
13251                                   <1>     ; !!! 'sysmsg' will put a 0 at ecx+1 position.!!!
13252 0001048A 51                       <1>     push    ecx
13253                                   <1>     ;;;
13254                                   <1>     ;
13255 0001048B 29CC                     <1>     sub     esp, ecx
13256 0001048D 89E7                     <1>     mov     edi, esp
13257 0001048F 89E6                     <1>     mov     esi, esp
13258 00010491 66891D[988E0100]         <1>     mov     [u.pcount], bx ; reset page (phy. addr.) counter
13259                                   <1>     ; 11/11/2015
13260 00010498 8A25[688E0100]           <1>     mov     ah, [u.ttyp] ; recent open tty
13261                                   <1>     ; 0 = none
13262 0001049E FECC                     <1>     dec     ah
13263 000104A0 790C                     <1>     jns     short sysmsg1
13264 000104A2 8A1D[858E0100]           <1>     mov     bl, [u.uno] ; process number
13265 000104A8 8AA3[478D0100]           <1>     mov     ah, [ebx+p.ttyc-1] ; user's (process's) console tty
13266                                   <1> sysmsg1:
13267 000104AE 8825[6A8E0100]           <1>     mov     [u.ttyn], ah
13268                                   <1> sysmsg2:
13269 000104B4 E8D2050000               <1>     call    cpass
13270 000104B9 7416                     <1>     jz      short sysmsg5
13271 000104BB AA                       <1>     stosb
13272 000104BC 20C0                     <1>     and     al, al
13273 000104BE 75F4                     <1>     jnz     short sysmsg2
13274                                   <1> sysmsg3:
13275 000104C0 80FC07                   <1>     cmp     ah, 7 ; tty number
13276 000104C3 7711                     <1>     ja      short sysmsg6 ; serial port
13277 000104C5 E83E000000               <1>     call    print_cmsg ; 05/12/2020
13278                                   <1> sysmsg4:
13279 000104CA 89EC                     <1>     mov     esp, ebp ; restore stack pointer
13280 000104CC E9ADC5FFFF               <1>     jmp     sysret
13281                                   <1> sysmsg5:
13282 000104D1 C60700                   <1>     mov     byte [edi], 0
13283 000104D4 EBEA                     <1>     jmp     short sysmsg3
13284                                   <1> sysmsg6:
13285 000104D6 8A06                     <1>     mov     al, [esi]
13286 000104D8 E8D5140000               <1>     call    sndc
13287 000104DD 72EB                     <1>     jc      short sysmsg4
13288 000104DF 803E00                   <1>     cmp     byte [esi], 0  ; 0 is stop character
13289 000104E2 76E6                     <1>     jna     short sysmsg4
13290 000104E4 46                       <1>     inc     esi
13291 000104E5 8A25[6A8E0100]           <1>     mov     ah, [u.ttyn]
13292 000104EB EBE9                     <1>     jmp     short sysmsg6
13293                                   <1>
13294                                   <1> sysmsgk: ; Temporary (01/07/2015)
13295                                   <1>     ; The message has been sent by Kernel (ASCIIZ string)
13296                                   <1>     ; (ECX -character count- will not be considered)
13297 000104ED 8B35[588E0100]           <1>     mov     esi, [u.base]
13298 000104F3 8A25[AE760100]           <1>     mov     ah, [ptty] ; present/current screen (video page)
13299 000104F9 8825[6A8E0100]           <1>     mov     [u.ttyn], ah
13300 000104FF C605[9A8E0100]00         <1>     mov     byte [u.kcall], 0
13301 00010506 EBB8                     <1>     jmp     short sysmsg3
13302                                   <1>
13303                                   <1> print_cmsg:
13304                                   <1>     ; 08/12/2020
13305                                   <1>     ; 07/12/2020
13306                                   <1>     ; 05/12/2020
13307                                   <1>     ; 18/11/2017
13308                                   <1>     ; 13/05/2016 - TRDOS 386 (TRDOS v2.0)
13309                                   <1>     ; 01/07/2015 (Retro UNIX 386 v1)
13310                                   <1>     ;
13311                                   <1>     ; print message (on user's console tty)
13312                                   <1>     ;       with requested color
13313                                   <1>     ;
13314                                   <1>     ; INPUTS:
13315                                   <1>     ;       esi = message address
13316                                   <1>     ;       [u.ttyn] = tty number (0 to 7)
13317                                   <1>     ;       [ccolor] = color attributes (IBM PC BIOS colors)
13318                                   <1>     ;
13319                                   <1>     ; Modified registers: eax, ebx, ecx, edx, esi, edi
13320                                   <1>     ; (ebp must be preserved)
13321                                   <1>
13322                                   <1>     ;mov    bh, ah
13323 00010508 8A3D[6A8E0100]           <1>     mov     bh, [u.ttyn]
13324 0001050E 8A1D[AF760100]           <1>     mov     bl, [ccolor] ; * ; 05/12/2020
13325                                   <1>
13326                                   <1>     ; 05/12/2020
13327 00010514 803D[EC9C0100]00         <1>     cmp     byte [pmi32], 0 ; is vbios's 32 bit pmi enabled ?
13328 0001051B 772E                     <1>     ja      short pcmsg5 ; yes
13329                                   <1> pcmsg1:
13330                                   <1>     ; 08/12/2020
13331 0001051D 8A1D[AF760100]           <1>     mov     bl, [ccolor] ; * (video.s 'u11'&'beep' change BL)
13332                                   <1>
13333 00010523 AC                       <1>     lodsb
13334 00010524 20C0                     <1>     and     al, al  ; 0
13335 00010526 743A                     <1>     jz      short pcmsg2
13336                                   <1> pcmsg7:
13337 00010528 56                       <1>     push    esi
13338                                   <1>     ;mov    bl, [ccolor] ; * (video.s 'u11'&'beep' change BL)
13339                                   <1>     ; 05/12/2020
13340                                   <1>     ;;mov   bh, [u.ttyn]
13341                                   <1>     ;call   _write_tty
13342                                   <1>     ;pop    esi
13343                                   <1>     ;jmp    short pcmsg1
13344                                   <1> ;pcmsg2:
13345                                   <1>     ;retn
13346                                   <1>
13347                                   <1>     ; 07/12/2020
13348 00010529 803D[1E680000]03         <1>     cmp     byte [CRT_MODE], 3
13349 00010530 7708                     <1>     ja      short pcmsg4
```

```
13350                                   <1> pcmsg3:
13351 00010532 E8B71DFFFF              <1>     call    _write_tty_m3
13352 00010537 5E                      <1>     pop     esi
13353 00010538 EBE3                    <1>     jmp     short pcmsg1
13354                                   <1> pcmsg4:
13355 0001053A 803D[1E680000]07        <1>         cmp     byte [CRT_MODE], 7
13356 00010541 76EF                    <1>     jna     short pcmsg3
13357 00010543 E86F2AFFFF              <1>     call    vga_write_teletype
13358 00010548 5E                      <1>     pop     esi
13359 00010549 EBD2                    <1>     jmp     short pcmsg1
13360                                   <1> pcmsg5:
13361                                   <1>     ; 07/12/2020
13362 0001054B 803D[1E680000]07        <1>         cmp     byte [CRT_MODE], 7
13363 00010552 76C9                    <1>     jna     short pcmsg1
13364                                   <1>
13365                                   <1>     ; 05/12/2020
13366                                   <1>     ; writing message by using
13367                                   <1>     ; VESA VBE3 video bios protected mode interface
13368                                   <1>
13369 00010554 B40E                    <1>     mov     ah, 0Eh
13370                                   <1> pcmsg6:
13371 00010556 AC                      <1>     lodsb
13372 00010557 20C0                    <1>     and     al, al  ; 0
13373 00010559 7407                    <1>     jz      short pcmsg2
13374                                   <1>     ; bh = video page
13375                                   <1>     ; ah = 0Eh
13376                                   <1>     ; al = character
13377                                   <1>     ; bl = color
13378 0001055B E8CF14FFFF              <1>     call    int10h_32bit_pmi
13379 00010560 EBF4                    <1>     jmp     short pcmsg6
13380                                   <1> pcmsg2:
13381 00010562 C3                      <1>     retn
13382                                   <1>
13383                                   <1> sysgeterr:
13384                                   <1>     ; 09/12/2015
13385                                   <1>     ; 21/09/2015 - (Retro UNIX 386 v1 feature only!)
13386                                   <1>     ; Get last error number or page fault count
13387                                   <1>     ; (for debugging)
13388                                   <1>     ;
13389                                   <1>     ; Input -> EBX = return type
13390                                   <1>     ;         0 = last error code (which is in 'u.error')
13391                                   <1>     ;         FFFFFFFFh = page fault count for running process
13392                                   <1>     ;         FFFFFFFEh = total page fault count
13393                                   <1>     ;         1 .. FFFFFFFDh = undefined
13394                                   <1>     ;
13395                                   <1>     ; Output -> EAX = last error number or page fault count
13396                                   <1>     ;         (depending on EBX input)
13397                                   <1>     ;
13398 00010563 21DB                    <1>     and     ebx, ebx
13399 00010565 750B                    <1>     jnz     short glerr_2
13400                                   <1> glerr_0:
13401 00010567 A1[A08E0100]            <1>     mov     eax, [u.error]
13402                                   <1> glerr_1:
13403 0001056C A3[348E0100]            <1>     mov     [u.r0], eax
13404 00010571 C3                      <1>     retn
13405                                   <1> glerr_2:
13406 00010572 43                      <1>     inc     ebx ; FFFFFFFFh -> 0, FFFFFFFEh -> FFFFFFFFh
13407 00010573 74FD                    <1>     jz      short glerr_2 ; page fault count for process
13408 00010575 43                      <1>     inc     ebx ; FFFFFFFFh -> 0
13409 00010576 75EF                    <1>     jnz     short glerr_0
13410 00010578 A1[50900100]            <1>     mov     eax, [PF_Count] ; total page fault count
13411 0001057D EBED                    <1>         jmp     short glerr_1
13412                                   <1> glerr_3:
13413 0001057F A1[A48E0100]            <1>     mov     eax, [u.pfcount]
13414 00010584 EBE6                    <1>     jmp     short glerr_1
13415                                   <1>
13416                                   <1> load_and_run_file:
13417                                   <1>     ; 07/07/2025 - TRDOS 386 Kernel v2.0.10
13418                                   <1>     ; 23/07/2022 - TRDOS 386 Kernel v2.0.5
13419                                   <1>     ; 18/11/2017
13420                                   <1>     ; 22/01/2017
13421                                   <1>     ; 04/01/2017 - 07/01/2017
13422                                   <1>     ; 24/10/2016
13423                                   <1>     ; 24/04/2016 - 02/05/2016 - 03/05/2016 - 06/05/2016
13424                                   <1>     ; 23/04/2016 (TRDOS 386 = TRDOS v2.0)
13425                                   <1>     ; 23/10/2015 (Retro UNIX 386 v1, 'sysexec')
13426                                   <1>     ; 23/06/2015 (Retro UNIX 386 v1 - Beginning)
13427                                   <1>     ; 03/06/2013 - 06/12/2013 (Retro UNIX 8086 v1)
13428                                   <1>     ; EAX = First Cluster number
13429                                   <1>     ; EDX = File Size
13430                                   <1>     ; ESI = Argument list address
13431                                   <1>     ; [argc] = argument count
13432                                   <1>     ; [u.nread] = argument list length
13433                                   <1>     ; [esp] = return address to the caller (*)
13434                                   <1>     ;
13435 00010586 8935[348F0100]          <1>     mov     [argv], esi
13436 0001058C 8915[408F0100]          <1>     mov     [i.size], edx
13437 00010592 A3[3C8F0100]            <1>     mov     [ii], eax
13438                                   <1>
13439                                   <1>     ;sti    ; 07/01/2017
13440                                   <1>     ;mov    eax, [k_page_dir]
13441                                   <1>     ;mov    [u.pgdir], eax
13442 00010597 31C0                    <1>     xor     eax, eax ; clc ; *** ; 04/01/2017
13443                                   <1>     ;mov    [u.r0], eax ; 0 ; 07/01/2017
13444                                   <1>
13445                                   <1>     ; 06/05/2016
13446                                   <1>     ; Set 'sysexit' return order to MainProg
13447                                   <1>     ;
13448 00010599 58                      <1>     pop     eax ; * 'loc_load_and_run_file_8:' address
13449                                   <1>     ;; 22/01/2017
13450                                   <1>     ;;cli ; 07/01/2017
13451 0001059A 8B25[1C760100]          <1>     mov     esp, [tss.esp0]
13452                                   <1>     ;
13453                                   <1>     ; 'loc_load_run_file_8' address has
13454                                   <1>     ; 'jmp loc_file_rw_restore_retn' instruction
13455                                   <1>     ; 'loc_file_rw_restore_retn:' will return to
13456                                   <1>     ; [mainprog_return_addr]
13457                                   <1>     ; just after 'call command_interpreter'
13458                                   <1>     ;
13459 000105A0 68[886D0000]            <1>     push    _end_of_mainprog ; we must not return to here !
13460 000105A5 FF35[00830100]          <1>     push    dword [mainprog_return_addr]
13461 000105AB 89E5                    <1>     mov     ebp, esp ; **
13462                                   <1>     ;
13463 000105AD 9C                      <1>     pushfd  ; EFLAGS      ; IRETD ; ***
13464 000105AE 6A08                    <1>     push    KCODE ; cs    ; IRETD
13465 000105B0 50                      <1>     push    eax ; * (eip) ; IRETD
13466 000105B1 8925[2C8E0100]          <1>     mov     [u.sp], esp
13467                                   <1>     ;mov    byte [u.quant], time_count
13468 000105B7 1E                      <1>     push    ds
13469 000105B8 06                      <1>     push    es
13470 000105B9 0FA0                    <1>     push    fs
13471 000105BB 0FA8                    <1>     push    gs
13472                                   <1>     ;mov    eax, [u.r0]
13473 000105BD 29C0                    <1>     sub     eax, eax
```

```
13474 000105BF 60                  <1>     pushad
13475 000105C0 68[7ECA0000]        <1>     push    sysret
13476                              <1>     ;push   sysrel1 ; 07/01/2017
13477 000105C5 8925[308E0100]      <1>     mov     [u.usp], esp
13478                              <1>     ;
13479 000105CB E8D6030000          <1>     call    wswap ; Save MainProg (process 1) 'u' structure
13480                              <1>                   ; and registers for return (from program)
13481 000105D0 89EC                <1>     mov     esp, ebp ; **
13482                              <1>     ;;22/01/2017
13483                              <1>     ;;sti ; 07/01/2017
13484                              <1>     ; 23/07/2022
13485                              <1>     ;push   eax  ; * 'loc_load_and_run_file_8:' address
13486                              <1>     ;
13487                              <1>     ;;; 02/05/2016
13488                              <1>     ;;; Create a new process (parent: MainProg)
13489 000105D2 31F6                <1>     xor     esi, esi
13490                              <1> cnpm_1: ; search p.stat table for unused process number
13491 000105D4 46                  <1>     inc     esi
13492 000105D5 80BE[678D0100]00    <1>     cmp     byte [esi+p.stat-1], 0 ; SFREE
13493                              <1>                            ; is process active, unused, dead
13494 000105DC 760B                <1>     jna     short cnpm_2   ; it's unused so branch
13495 000105DE 6683FE10            <1>     cmp     si, nproc      ; all processes checked
13496 000105E2 72F0                <1>     jb      short cnpm_1    ; no, branch back
13497                              <1> cnpm_panic:
13498 000105E4 E92068FFFF          <1>     jmp     panic
13499                              <1> cnpm_2:
13500 000105E9 A1[8C8E0100]        <1>     mov     eax, [u.pgdir] ; page directory of MainProg
13501 000105EE A3[908E0100]        <1>     mov     [u.ppgdir], eax ; parent's page directory
13502 000105F3 E82E52FFFF          <1>     call    allocate_page
13503                              <1>     ;jc     panic
13504                              <1>     ; 23/07/2022
13505 000105F8 72EA                <1>     jc      short cnpm_panic
13506                              <1>
13507                              <1>     ; EAX = UPAGE (user structure page) address
13508 000105FA A3[888E0100]        <1>     mov     [u.upage], eax ; memory page for 'user' struct (child)
13509 000105FF 89F7                <1>     mov     edi, esi
13510                              <1>     ;shl    di, 2
13511                              <1>     ; 23/07/2022
13512 00010601 C1E702              <1>     shl     edi, 2
13513 00010604 8987[748D0100]      <1>     mov     [edi+p.upage-4], eax ; memory page for 'user' struct
13514 0001060A E88852FFFF          <1>     call    clear_page ; 03/05/2016
13515                              <1>     ;;movzx eax, byte [p.ttyc] ; console tty (for MainProg)
13516                              <1>     ;sub    ax, ax ; 0
13517                              <1>     ; 23/07/2022
13518 0001060F 29C0                <1>     sub     eax, eax
13519 00010611 668986[478D0100]    <1>     mov     [esi+p.ttyc-1], ax ; al - set child's console tty
13520                              <1>                                ; ah - reset child's wait channel
13521 00010618 66A3[688E0100]      <1>     mov     [u.ttyp], ax ; 0
13522                              <1>
13523 0001061E 89F2                <1>     mov     edx, esi
13524 00010620 8815[858E0100]      <1>     mov     [u.uno], dl ; child process number
13525 00010626 FE86[678D0100]      <1>         inc     byte [esi+p.stat-1] ; 1, SRUN
13526                              <1>     ;shl    si, 1 ; multiply si by 2 to get index into p.pid table
13527                              <1>     ; 23/07/2022
13528 0001062C D1E6                <1>     shl     esi, 1
13529 0001062E 66FF05[1C8E0100]    <1>     inc     word [mpid] ; increment m.pid; get a new process name
13530                              <1>
13531                              <1>     ; 23/07/2022
13532                              <1>     ;mov    ax, [p.pid]  ; get process name of MainProg
13533                              <1>     ;mov    ax, 1
13534 00010635 FEC0                <1>     inc     al ; eax = 1
13535 00010637 668986[268D0100]    <1>     mov     [esi+p.ppid-2], ax ; put parent process name
13536                              <1>                                ; in parent process slot for child
13537 0001063E A2[7E8E0100]        <1>     mov     [u.pri], al ; 1       ; normal priority
13538                              <1>
13539                              <1>     ;dec    ax ; 0
13540                              <1>     ;mov    [u.ttyp], ax ; 0
13541                              <1>
13542 00010643 66A1[1C8E0100]      <1>     mov     ax, [mpid]
13543 00010649 668986[068D0100]    <1>     mov     [esi+p.pid-2], ax ; put new process name
13544                              <1>                               ; in child process' name slot
13545                              <1>     ;;;
13546                              <1>     ; 07/07/2025
13547                              <1>     ;mov    eax, [ii]
13548                              <1>     ; 23/07/2022
13549                              <1>     ; Retro UNIX 386 v1, 'sysexec' (u2.s)
13550                              <1>     ;call   iopen
13551                              <1>     ; 06/06/2016
13552                              <1>     ;mov    byte [u.pri], 1 ; normal priority
13553                              <1>     ;
13554                              <1>     ;jmp    short sysexec_7 ; 02/05/2016
13555                              <1>     ; 23/07/2022
13556 00010650 E943F9FFFF          <1>     jmp     sysexec_7
13557                              <1>
13558                              <1> ;   ; 02/05/2016
13559                              <1> ;   ;inc    byte [sysflg] ; 0FFh -> 0
13560                              <1> ;   ;mov    byte [sysflg], 0 ; 04/01/2017
13561                              <1> ;   movzx   ebx, byte [u.uno]
13562                              <1> ;   shl     bl, 1 ; 13/11/2017
13563                              <1> ;   cmp     word [ebx+p.ppid-2], 1 ; MainProg
13564                              <1> ;   ja      sysret0 ; 03/05/2016
13565                              <1> ;   push    sysret ; *
13566                              <1> ;   mov     [u.usp], esp
13567                              <1> ;   call    wswap ; save child process 'u' structure and
13568                              <1> ;                 ; registers
13569                              <1> ;   add     dword [u.usp], 4 ; 03/05/2016
13570                              <1> ;sysexec_19: ; 02/05/2016
13571                              <1> ;   retn ; * 'sysret' ; byte [sysflg] -> 0FFh
13572                              <1>
13573                              <1> readi:
13574                              <1>     ; 09/08/2022
13575                              <1>     ; 23/07/2022 - TRDOS 386 Kernel v2.0.5
13576                              <1>     ; 01/05/2016
13577                              <1>     ; 25/04/2016 - TRDOS 386 (TRDOS v2.0)
13578                              <1>     ; 20/05/2015 - Retro UNIX 386 v1
13579                              <1>     ; 11/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
13580                              <1>     ;
13581                              <1>     ; Reads from a file whose the first cluster number in EAX
13582                              <1>     ;
13583                              <1>     ; INPUTS ->
13584                              <1>     ;    EAX - First cluster number of the file
13585                              <1>     ;    u.count - byte count user desires
13586                              <1>     ;    u.base - points to user buffer
13587                              <1>     ;    u.fofp - points to dword with current file offset
13588                              <1>     ;    i.size - file size
13589                              <1>     ;    cdev - logical dos drive number of the file
13590                              <1>     ; OUTPUTS ->
13591                              <1>     ;    u.count - cleared
13592                              <1>     ;    u.nread - accumulates total bytes passed back
13593                              <1>     ;
13594                              <1>     ; ((EAX)) input/output
13595                              <1>     ; (Retro UNIX Prototype : 14/12/2012 - 01/03/2013, UNIXCOPY.ASM)
13596                              <1>       ; ((Modified registers: edx, ebx, ecx, esi, edi))
13597                              <1>
```

```
13598 00010655 31D2                <1>     xor     edx, edx ; 0
13599 00010657 8915[608E0100]      <1>     mov     [u.nread], edx ; 0
13600 0001065D 668915[988E0100]    <1>     mov     [u.pcount], dx ; 19/05/2015
13601 00010664 3915[5C8E0100]      <1>     cmp     [u.count], edx ; 0
13602                              <1>     ;ja     short readi_1
13603                              <1>     ;retn
13604                              <1>     ; 09/08/2022
13605 0001066A 765E                <1>     jna     short dskr_5 ; retn
13606                              <1> ;readi_1:
13607                              <1> dskr:
13608                              <1>     ; 01/05/2016
13609                              <1>     ; 25/04/2016 - TRDOS 386 (TRDOS v2.0)
13610                              <1>     ; 24/05/2015 - 12/10/2015 (Retro UNIX 386 v1)
13611                              <1>     ; 26/04/2013 - 03/08/2013 (Retro UNIX 8086 v1)
13612                              <1> dskr_0:
13613 0001066C 8B15[408F0100]      <1>         mov edx, [i.size]
13614 00010672 8B1D[488E0100]      <1>     mov     ebx, [u.fofp]
13615 00010678 2B13                <1>     sub     edx, [ebx]
13616 0001067A 7647                <1>     jna     short dskr_4
13617                              <1>     ;
13618 0001067C 50                  <1>     push    eax ; 01/05/2016
13619 0001067D 3B15[5C8E0100]      <1>     cmp     edx, [u.count]
13620 00010683 7306                <1>     jnb     short dskr_1
13621 00010685 8915[5C8E0100]      <1>     mov     [u.count], edx
13622                              <1> dskr_1:
13623                              <1>     ; EAX = First Cluster
13624                              <1>     ; [Current_Drv] = Physical drive number
13625 0001068B E83B000000          <1>     call    mget_r
13626                              <1>     ; NOTE: in 'mget_r', relevant sector will be read in buffer
13627                              <1>     ; if it is not already in buffer !
13628 00010690 BB[5C900100]        <1>     mov     ebx, readi_buffer
13629 00010695 803D[9A8E0100]00    <1>     cmp     byte [u.kcall], 0 ; the caller is 'namei' sign (=1)
13630 0001069C 770F                <1>     ja      short dskr_3     ; zf=0 -> the caller is 'namei'
13631 0001069E 66833D[988E0100]00  <1>     cmp     word [u.pcount], 0
13632 000106A6 7705                <1>     ja      short dskr_3
13633                              <1> dskr_2:
13634                              <1>     ; [u.base] = virtual address to transfer (as destination address)
13635 000106A8 E869010000          <1>     call    trans_addr_w ; translate virtual address to physical (w)
13636                              <1> dskr_3:
13637                              <1>     ; EBX (r5) = system (I/O) buffer address -physical-
13638 000106AD E8CC010000          <1>     call    sioreg
13639 000106B2 87FE                <1>     xchg    esi, edi
13640                              <1>     ; EDI = file (user data) offset
13641                              <1>     ; ESI = sector (I/O) buffer offset
13642                              <1>     ; ECX = byte count
13643 000106B4 F3A4                <1>     rep     movsb
13644                              <1>     ; eax = remain bytes in buffer
13645                              <1>       ;       (check if remain bytes in the buffer > [u.pcount])
13646 000106B6 09C0                <1>     or      eax, eax
13647 000106B8 75EE                <1>     jnz     short dskr_2 ; (page end before system buffer end!)
13648 000106BA 58                  <1>     pop     eax  ; (first cluster number)
13649 000106BB 390D[5C8E0100]      <1>     cmp     [u.count], ecx ; 0
13650 000106C1 77A9                <1>     ja      short dskr_0
13651                              <1> dskr_4:
13652 000106C3 C605[9A8E0100]00    <1>     mov     byte [u.kcall], 0
13653                              <1> dskr_5:             ; 23/07/2022
13654 000106CA C3                  <1>     retn
13655                              <1>
13656                              <1> mget_r:
13657                              <1>     ; 19/12/2025 - TRDOS 386 Kernel v2.0.10
13658                              <1>     ; 29/08/2023
13659                              <1>     ; 30/07/2022
13660                              <1>     ; 23/07/2022 - TRDOS 386 Kernel v2.0.5
13661                              <1>     ; 24/10/2016
13662                              <1>     ; 22/10/2016
13663                              <1>     ; 12/10/2016
13664                              <1>     ; 29/04/2016
13665                              <1>     ; 25/04/2016 - TRDOS 386 (TRDOS v2.0)
13666                              <1>     ; 03/06/2015 (Retro UNIX 386 v1, 'mget', u.5s)
13667                              <1>     ; 22/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
13668                              <1>     ;
13669                              <1>     ; Get existing or (allocate) a new disk block for file
13670                              <1>     ;
13671                              <1>     ; INPUTS ->
13672                              <1>     ;    [u.fofp] = file offset pointer
13673                              <1>     ;    EAX = First Cluster
13674                              <1>     ;    [cdev] = Logical dos drive number
13675                              <1>     ;    ([u.off] = file offset)
13676                              <1>     ; OUTPUTS ->
13677                              <1>     ;    EAX = logical sector number
13678                              <1>     ;    ESI = Logical Dos Drive Description Table address
13679                              <1>     ;
13680                              <1>     ; Modified registers: EDX, EBX, ECX, ESI, EDI
13681                              <1>
13682 000106CB 8B35[488E0100]      <1>     mov     esi, [u.fofp]
13683 000106D1 8B1E                <1>     mov     ebx, [esi] ; (u.off)
13684                              <1>
13685 000106D3 29C9                <1>     sub     ecx, ecx
13686 000106D5 8A2D[198E0100]      <1>     mov     ch, [cdev]
13687                              <1>
13688 000106DB BE00010900          <1>     mov     esi, Logical_DOSDisks
13689 000106E0 01CE                <1>     add     esi, ecx
13690                              <1>
13691 000106E2 380D[B0820100]      <1>     cmp     [readi.valid], cl ; 0
13692 000106E8 7642                <1>     jna     short mget_r_0
13693                              <1>
13694 000106EA 3A2D[B1820100]      <1>     cmp     ch, [readi.drv]
13695 000106F0 753A                <1>     jne     short mget_r_0
13696                              <1>
13697 000106F2 3B05[C4820100]      <1>     cmp     eax, [readi.fclust]
13698 000106F8 754E                <1>     jne     short mget_r_3
13699                              <1>
13700 000106FA 89D8                <1>     mov     eax, ebx ; file offset
13701 000106FC 668B0D[B8820100]    <1>     mov     cx, [readi.bpc]
13702 00010703 41                  <1>     inc     ecx ; <= 65536
13703 00010704 29D2                <1>     sub     edx, edx
13704 00010706 F7F1                <1>     div     ecx
13705                              <1>
13706 00010708 8B3D[C0820100]      <1>     mov     edi, [readi.c_index] ; cluster index
13707                              <1>
13708 0001070E 39F8                <1>     cmp     eax, edi
13709 00010710 7563                <1>         jne     short mget_r_4  ; (*)
13710                              <1>
13711                              <1>     ; edx = byte offset in cluster (<= 65535)
13712 00010712 668915[BA820100]    <1>     mov     [readi.offset], dx
13713                              <1>     ; 23/07/2022
13714                              <1>     ;shr    dx, 9 ; / 512
13715 00010719 C1EA09              <1>     shr     edx, 9
13716 0001071C 8815[B3820100]      <1>     mov     [readi.s_index], dl ; sector index in cluster (0 to spc -1)
13717                              <1>
13718 00010722 A1[BC820100]        <1>     mov     eax, [readi.cluster]  ; > 0 if [readi.valid] = 1
13719                              <1>     ; 19/12/2025
13720                              <1>     ;mov    edx, [readi.fs_index]
13721 00010727 E989000000          <1>         jmp     mget_r_7
```

```
13722                                   <1>
13723                                   <1> mget_r_0:
13724 0001072C 882D[B1820100]           <1>     mov     [readi.drv], ch ; physical drive number
13725                                   <1> ; 19/12/2025
13726                                   <1> %if 0
13727                                   <1>     cmp     byte [esi+LD_FATType], 0
13728                                   <1>     ja      short mget_r_1
13729                                   <1>     ;mov    cl, [esi+LD_FS_BytesPerSec+1]
13730                                   <1>     ;shr    cl, 1   ; 1 for 512 bytes, 4 for 2048 bytes
13731                                   <1>     ; 19/12/2025
13732                                   <1>     mov     cl, 1
13733                                   <1>     jmp     short mget_r_2
13734                                   <1> %endif
13735                                   <1> mget_r_1:
13736 00010732 8A4E13                   <1>     mov     cl, [esi+LD_BPB+BPB_SecPerClust]
13737                                   <1> mget_r_2:
13738 00010735 880D[B2820100]           <1>     mov     [readi.spc], cl  ; sectors per cluster
13739                                   <1>     ; NOTE: readi bytes per sector value is always 512 !
13740                                   <1>     ; 23/07/2022
13741                                   <1>     ;xor    ch, ch
13742                                   <1>     ; 29/08/2023
13743 0001073B C1E109                   <1>     shl     ecx, 9
13744                                   <1>     ;shl    cx, 9 ; * 512
13745                                   <1>     ;dec    cx ; bytes per cluster - 1
13746                                   <1>     ; 23/07/2022
13747 0001073E 49                       <1>     dec     ecx
13748 0001073F 66890D[B8820100]         <1>     mov     [readi.bpc], cx
13749                                   <1>     ;sub    cx, cx
13750                                   <1>     ; 23/07/2022
13751 00010746 29C9                     <1>     sub     ecx, ecx
13752                                   <1> mget_r_3:
13753 00010748 A3[C4820100]             <1>     mov     [readi.fclust], eax ; first cluster (or FDT address)
13754 0001074D 880D[B0820100]           <1>     mov     [readi.valid], cl ; 0
13755                                   <1>     ;mov    [readi.s_index], cl ; 0
13756                                   <1>     ;mov    [readi.offset], cx ; 0
13757 00010753 890D[C0820100]           <1>     mov     [readi.c_index], ecx ; 0
13758 00010759 890D[BC820100]           <1>     mov     [readi.cluster], ecx ; 0
13759 0001075F 890D[B4820100]           <1>     mov     [readi.sector], ecx ; 0
13760                                   <1>
13761 00010765 89D8                     <1>     mov     eax, ebx ; file offset
13762 00010767 668B0D[B8820100]         <1>     mov     cx, [readi.bpc]
13763 0001076E 41                       <1>     inc     ecx ; <= 65536
13764 0001076F 29D2                     <1>     sub     edx, edx
13765 00010771 F7F1                     <1>     div     ecx
13766                                   <1>     ;mov    edi, [readi.c_index] ; previous cluster index
13767 00010773 29FF                     <1>     sub     edi, edi
13768                                   <1> mget_r_4:
13769 00010775 A3[C0820100]             <1>     mov     [readi.c_index], eax ; cluster index
13770                                   <1>     ; edx = byte offset in cluster (<= 65535)
13771 0001077A 668915[BA820100]         <1>     mov     [readi.offset], dx
13772                                   <1>     ; 23/07/2022
13773                                   <1>     ;shr    dx, 9 ; / 512
13774 00010781 C1EA09                   <1>     shr     edx, 9
13775 00010784 8815[B3820100]           <1>     mov     [readi.s_index], dl ; sector index in cluster (0 to spc -1)
13776                                   <1>
13777 0001078A 89C1                     <1>     mov     ecx, eax ; current cluster index
13778 0001078C A1[C4820100]             <1>     mov     eax, [readi.fclust]
13779 00010791 09C9                     <1>     or      ecx, ecx ; cluster index
13780 00010793 741B                     <1>     jz      short mget_r_6
13781                                   <1>
13782 00010795 39CF                     <1>     cmp     edi, ecx
13783 00010797 7710                     <1>     ja      short mget_r_5 ; old cluster index is higher
13784 00010799 8B15[BC820100]           <1>     mov     edx, [readi.cluster]
13785 0001079F 21D2                     <1>     and     edx, edx
13786 000107A1 7406                     <1>     jz      short mget_r_5
13787                                   <1>     ; valid 'readi' parameters (*)
13788 000107A3 89D0                     <1>     mov     eax, edx
13789 000107A5 29F9                     <1>     sub     ecx, edi
13790 000107A7 740C                     <1>     jz      short mget_r_7
13791                                   <1> mget_r_5:
13792                                   <1>     ; EAX = Beginning cluster
13793                                   <1>     ; EDX = Sector index in disk/file section
13794                                   <1>     ;       (Only for SINGLIX file system!)
13795                                   <1>     ; ECX = Cluster sequence number after the beginning cluster
13796                                   <1>     ; ESI = Logical DOS Drive Description Table address
13797 000107A9 E8FBC0FFFF               <1>     call    get_cluster_by_index
13798 000107AE 7246                     <1>     jc      short mget_r_err
13799                                   <1>     ; EAX = Cluster number
13800                                   <1> mget_r_6:
13801 000107B0 A3[BC820100]             <1>     mov     [readi.cluster], eax ; FDT number for Singlix File System
13802                                   <1> mget_r_7:
13803                                   <1> ; 19/12/2025
13804                                   <1> %if 0
13805                                   <1>     cmp     byte [esi+LD_FATType], 0
13806                                   <1>     jna     short mget_r_12
13807                                   <1> %endif
13808                                   <1>     ;sub    eax, 2
13809                                   <1>     ; 30/07/2022
13810 000107B5 48                       <1>     dec     eax
13811 000107B6 48                       <1>     dec     eax
13812 000107B7 0FB615[B2820100]         <1>     movzx   edx, byte [readi.spc]
13813 000107BE F7E2                     <1>     mul     edx
13814                                   <1>
13815 000107C0 034668                   <1>     add     eax, [esi+LD_DATABegin]
13816 000107C3 8A15[B3820100]           <1>     mov     dl, [readi.s_index]
13817 000107C9 01D0                     <1>     add     eax, edx
13818                                   <1> mget_r_8:
13819                                   <1>     ; eax = logical sector number
13820 000107CB 803D[B0820100]00         <1>     cmp     byte [readi.valid], 0
13821 000107D2 7608                     <1>     jna     short mget_r_9
13822 000107D4 3B05[B4820100]           <1>     cmp     eax, [readi.sector]
13823 000107DA 7435                     <1>     je      short mget_r_11 ; sector is already in 'readi' buffer
13824                                   <1> mget_r_9:
13825 000107DC A3[B4820100]             <1>     mov     [readi.sector], eax
13826 000107E1 BB[5C900100]             <1>     mov     ebx, readi_buffer ; buffer address
13827                                   <1>     ;mov    ecx, 1
13828                                   <1>     ; 30/07/2022
13829 000107E6 31C9                     <1>     xor     ecx, ecx
13830 000107E8 FEC1                     <1>     inc     cl
13831                                   <1>     ; ecx = 1
13832                                   <1>
13833                                   <1>     ; 29/04/2016
13834                                   <1>     ;xor    dl, dl
13835                                   <1>
13836                                   <1>     ; EAX = Logical sector number
13837                                   <1>     ; ECX = Sector count
13838                                   <1>     ; EBX = Buffer address
13839                                   <1>     ; (EDX = 0)
13840                                   <1>     ; ESI = Logical DOS drive description table address
13841                                   <1>
13842 000107EA E8D3110000               <1>     call    disk_read
13843 000107EF 7314                     <1>     jnc     short mget_r_10
13844                                   <1>
13845                                   <1>     ; 22/10/2016 (15h -> 17)
```

```
13846 000107F1 B811000000          <1>     mov     eax, 17 ; Drive not ready or read error !
13847                              <1> mget_r_err:
13848 000107F6 A3[A08E0100]        <1>     mov     [u.error], eax
13849                              <1>     ; 12/10/2016
13850 000107FB A3[348E0100]        <1>     mov     [u.r0], eax
13851 00010800 E959C2FFFF          <1>     jmp     error
13852                              <1> mget_r_10:
13853 00010805 C605[B0820100]01    <1>     mov     byte [readi.valid], 1 ; 24/10/2016
13854 0001080C A1[B4820100]        <1>     mov     eax, [readi.sector]
13855                              <1> mget_r_11:
13856 00010811 C3                  <1>     retn
13857                              <1>
13858                              <1> ; 19/12/2025
13859                              <1> %if 0
13860                              <1> mget_r_12:
13861                              <1>     ; EAX = FDT number
13862                              <1>     ; EDX = Sector index from FDT sector (0,1,2,3,4...)
13863                              <1>     inc     eax ; the first data sector in FS disk section
13864                              <1>     mov     [readi.fs_index], edx
13865                              <1>     add     eax, edx
13866                              <1>     jmp     short mget_r_8
13867                              <1> %endif
13868                              <1>
13869                              <1> trans_addr_r:
13870                              <1>     ; 12/10/2016
13871                              <1>     ; 02/05/2016 - TRDOS 386 (TRDOS v2.0)
13872                              <1>     ; Translate virtual address to physical address
13873                              <1>     ; for reading from user's memory space
13874                              <1>     ; 04/06/2015 - 18/10/2015 (Retro UNIX 386 v1)
13875                              <1>
13876 00010812 31D2                <1>     xor     edx, edx ; 0 (read access sign)
13877 00010814 EB04                <1>     jmp     short trans_addr_rw
13878                              <1>
13879                              <1> trans_addr_w:
13880                              <1>     ; 12/10/2016
13881                              <1>     ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
13882                              <1>     ; Translate virtual address to physical address
13883                              <1>     ; for writing to user's memory space
13884                              <1>     ; 04/06/2015 - 18/10/2015 (Retro UNIX 386 v1)
13885                              <1>
13886 00010816 29D2                <1>     sub     edx, edx
13887 00010818 FEC2                <1>     inc     dl ; 1 (write access sign)
13888                              <1> trans_addr_rw:
13889 0001081A 50                  <1>     push    eax
13890 0001081B 53                  <1>     push    ebx
13891 0001081C 52                  <1>     push    edx ; r/w sign (in DL)
13892                              <1>     ;
13893 0001081D 8B1D[588E0100]      <1>     mov     ebx, [u.base]
13894 00010823 E8DC53FFFF          <1>     call    get_physical_addr ; get physical address
13895 00010828 730F                <1>     jnc     short passc_0
13896 0001082A A3[A08E0100]        <1>     mov     [u.error], eax
13897 0001082F A3[348E0100]        <1>     mov     [u.r0], eax ; 12/10/2016
13898                              <1>     ;pop    edx
13899                              <1>     ;pop    ebx
13900                              <1>     ;pop    eax
13901 00010834 E925C2FFFF          <1>     jmp     error
13902                              <1> passc_0:
13903 00010839 F6C202              <1>     test    dl, PTE_A_WRITE ; writable page
13904 0001083C 5A                  <1>     pop     edx
13905 0001083D 751C                <1>     jnz     short passc_1
13906                              <1>
13907 0001083F 20D2                <1>     and     dl, dl
13908 00010841 7418                <1>     jz      short passc_1
13909                              <1>     ; read only (duplicated) page -must be copied to a new page-
13910                              <1>     ; EBX = linear address
13911 00010843 51                  <1>     push    ecx
13912 00010844 E82C53FFFF          <1>     call    copy_page
13913 00010849 59                  <1>     pop     ecx
13914 0001084A 721E                <1>     jc      short passc_2
13915 0001084C 50                  <1>     push    eax ; physical address of the new/allocated page
13916 0001084D E8B253FFFF          <1>     call    add_to_swap_queue
13917 00010852 58                  <1>     pop     eax
13918 00010853 81E3FF0F0000        <1>     and     ebx, PAGE_OFF ; 0FFFh
13919                              <1>     ;mov    ecx, PAGE_SIZE
13920                              <1>     ;sub    ecx, ebx
13921 00010859 01D8                <1>     add     eax, ebx
13922                              <1> passc_1:
13923 0001085B A3[948E0100]        <1>     mov     [u.pbase], eax ; physical address
13924 00010860 66890D[988E0100]    <1>     mov     [u.pcount], cx ; remain byte count in page (1-4096)
13925 00010867 5B                  <1>     pop     ebx
13926 00010868 58                  <1>     pop     eax
13927 00010869 C3                  <1>     retn
13928                              <1> passc_2:
13929 0001086A B804000000          <1>     mov     eax, ERR_MINOR_IM ; "Insufficient memory !" error
13930 0001086F A3[348E0100]        <1>     mov     [u.r0], eax ; 12/10/2016
13931 00010874 A3[A08E0100]        <1>     mov     [u.error], eax
13932                              <1>     ;pop    ebx
13933                              <1>     ;pop    eax
13934 00010879 E9E0C1FFFF          <1>     jmp     error
13935                              <1>
13936                              <1> sioreg:
13937                              <1>     ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
13938                              <1>     ; 19/05/2015 - 25/07/2015 (Retro UNIX 386 v1)
13939                              <1>     ; 12/03/2013 - 22/07/2013 (Retro UNIX 8086 v1)
13940                              <1>     ; INPUTS ->
13941                              <1>     ;     EBX = system buffer (data) address (r5)
13942                              <1>     ;     [u.fofp] = pointer to file offset pointer
13943                              <1>     ;     [u.base] = virtual address of the user buffer
13944                              <1>     ;     [u.pbase] = physical address of the user buffer
13945                              <1>     ;     [u.count] = byte count
13946                              <1>     ;     [u.pcount] = byte count within page frame
13947                              <1>     ; OUTPUTS ->
13948                              <1>     ;     ESI = user data offset (r1)
13949                              <1>     ;     EDI = system (I/O) buffer offset (r2)
13950                              <1>     ;     ECX = byte count (r3)
13951                              <1>     ;     EAX = remain bytes after byte count within page frame
13952                              <1>     ;       (If EAX > 0, transfer will continue from the next page)
13953                              <1>         ;
13954                              <1>     ; ((Modified registers:  EDX))
13955                              <1>
13956 0001087E 8B35[488E0100]      <1>         mov     esi, [u.fofp]
13957 00010884 8B3E                <1>         mov     edi, [esi]
13958 00010886 89F9                <1>     mov     ecx, edi
13959 00010888 81C900FEFFFF        <1>     or      ecx, 0FFFFFE00h
13960 0001088E 81E7FF010000        <1>     and     edi, 1FFh
13961 00010894 01DF                <1>     add     edi, ebx ; EBX = system buffer (data) address
13962 00010896 F7D9                <1>     neg     ecx
13963 00010898 3B0D[5C8E0100]      <1>     cmp     ecx, [u.count]
13964 0001089E 7606                <1>     jna     short sioreg_0
13965 000108A0 8B0D[5C8E0100]      <1>     mov     ecx, [u.count]
13966                              <1> sioreg_0:
13967 000108A6 803D[9A8E0100]00    <1>     cmp     byte [u.kcall], 0
13968 000108AD 7613                <1>     jna     short sioreg_1
13969                              <1>      ; the caller is 'mkdir' or 'namei'
```

```
13970 000108AF A1[588E0100]        <1>     mov     eax, [u.base]
13971 000108B4 A3[948E0100]        <1>     mov     [u.pbase], eax ; physical address = virtual address
13972 000108B9 66890D[988E0100]    <1>     mov     word [u.pcount], cx ; remain bytes in buffer (1 sector)
13973 000108C0 EB0B                <1>     jmp     short sioreg_2
13974                              <1> sioreg_1:
13975 000108C2 0FB715[988E0100]    <1>     movzx   edx, word [u.pcount]
13976 000108C9 39D1                <1>     cmp     ecx, edx
13977 000108CB 772A                <1>     ja      short sioreg_4 ; transfer count > [u.pcount]
13978                              <1> sioreg_2: ; 2:
13979 000108CD 31C0                <1>     xor     eax, eax
13980                              <1> sioreg_3:
13981 000108CF 010D[608E0100]      <1>     add     [u.nread], ecx
13982 000108D5 290D[5C8E0100]      <1>     sub     [u.count], ecx
13983 000108DB 010D[588E0100]      <1>     add     [u.base], ecx
13984 000108E1 010E                <1>         add         [esi], ecx
13985 000108E3 8B35[948E0100]      <1>     mov     esi, [u.pbase]
13986 000108E9 66290D[988E0100]    <1>     sub     [u.pcount], cx
13987 000108F0 010D[948E0100]      <1>     add     [u.pbase], ecx
13988 000108F6 C3                  <1>         retn
13989                              <1> sioreg_4:
13990                              <1>     ; transfer count > [u.pcount]
13991                              <1>     ; (ecx > edx)
13992 000108F7 89C8                <1>     mov     eax, ecx
13993 000108F9 29D0                <1>     sub     eax, edx ; remain bytes for 1 sector (block) transfer
13994 000108FB 89D1                <1>     mov     ecx, edx ; current transfer count = [u.pcount]
13995 000108FD EBD0                <1>     jmp     short sioreg_3
13996                              <1>
13997                              <1> tswitch: ; Retro UNIX 386 v1
13998                              <1> tswap:
13999                              <1>     ; 16/01/2017
14000                              <1>     ; 21/05/2016 - TRDOS 386 (TRDOS v2.0)
14001                              <1>     ; 10/05/2015 - 01/09/2015 (Retro UNIX 386 v1)
14002                              <1>     ; 14/04/2013 - 14/02/2014 (Retro UNIX 8086 v1)
14003                              <1>     ; time out swap, called when a user times out.
14004                              <1>     ; the user is put on the low priority queue.
14005                              <1>     ; This is done by making a link from the last user
14006                              <1>     ; on the low priority queue to him via a call to 'putlu'.
14007                              <1>     ; then he is swapped out.
14008                              <1>
14009                              <1>     ; TRDOS 386 (TRDOS v2.0) modification ->  ** 21/05/2016 **
14010                              <1>     ;     * when a high priority (event) process will be stopped
14011                              <1>     ;       (swapped out, swithched out/off), 'tswap/tswitch' will
14012                              <1>     ;       not add it to a run queue.
14013                              <1>     ;       /// What for: Process may be already in a run queue,
14014                              <1>     ;       it is unspeficied state because process might be started
14015                              <1>     ;       by a timer event which does not regard previous priority
14016                              <1>     ;       level and run queue of the process (for fast executing!).
14017                              <1>     ;       After the 'run for event', process will be sequenced
14018                              <1>     ;       to run by it's actual run queue. ///
14019                              <1>     ;
14020                              <1>     ; Retro UNIX 386 v1 modification ->
14021                              <1>     ;       swap (software task switch) is performed by changing
14022                              <1>     ;       user's page directory (u.pgdir) instead of segment change
14023                              <1>     ;       as in Retro UNIX 8086 v1.
14024                              <1>     ;
14025                              <1>     ; RETRO UNIX 8086 v1 modification ->
14026                              <1>     ;       'swap to disk' is replaced with 'change running segment'
14027                              <1>     ;       according to 8086 cpu (x86 real mode) architecture.
14028                              <1>     ;       pdp-11 was using 64KB uniform memory while IBM PC
14029                              <1>     ;       compatibles was using 1MB segmented memory
14030                              <1>     ;       in 8086/8088 times.
14031                              <1>     ;
14032                              <1>     ; INPUTS ->
14033                              <1>     ;    u.uno - users process number
14034                              <1>     ;    runq+4 - lowest priority queue
14035                              <1>     ; OUTPUTS ->
14036                              <1>     ;    r0 - users process number
14037                              <1>     ;    r2 - lowest priority queue address
14038                              <1>     ;
14039                              <1>     ; ((AX = R0, BX = R2)) output
14040                              <1>     ; ((Modified registers: EDX, EBX, ECX, ESI, EDI))
14041                              <1>     ;
14042                              <1>
14043                              <1>     NOTE:
14044                              <1>     ;* [u.pri] priority level is specified by run queue which is process
14045                              <1>     ;  comes to run from.
14046                              <1>     ;* Initial [u.pri] is 1 ('normal/regular') for programs
14047                              <1>     ;  (which are launched by MainProg or 'sysexec'), it is changed
14048                              <1>     ;  to 2 ('high') by timer event, if program uses 'systimer' system call.
14049                              <1>     ;* Program (Process) also can change it's running priority
14050                              <1>     ;  from 1 to 0 or up to 2 by using 'syspri' system call; but,
14051                              <1>     ;  if program selects priority level 2 (high) for running, next time
14052                              <1>     ;  it is reduced to 1 (normal/regular) because 'syspri' adds this
14053                              <1>     ;  program to 'run for normal' queue while running duration is a bit
14054                              <1>     ;  protected from swap/switch out immediate, behalf of other high
14055                              <1>     ;  priority process in sequence. Program (with high priority) will not
14056                              <1>     ;  be swapped/switched out (by timer event) before it's time quantum
14057                              <1>     ;  will be elapsed, but, this will be temporary if program is not using
14058                              <1>     ;  timer event function.
14059                              <1>
14060                              <1>     ;For example:
14061                              <1>     ;If a process frequently gets a timer event, it runs at high priority
14062                              <1>     ;level but when it returns from running it returns to actual run queue,
14063                              <1>     ;not to 'run for event' queue again.
14064                              <1>     ;'tswap' will not change the sequence at return/stop(swap out) stage.
14065                              <1>     ;But if priority level not high (=2, 'run for event'), 'tswap/tswitch'
14066                              <1>     ;will add the stopping process to relevant run queue according to
14067                              <1>     ;[u.pri] priority level.
14068                              <1>
14069                              <1>     ; 16/01/2017
14070 000108FF BB[248E0100]        <1>     mov     ebx, runq+2    ; 'runq_normal' ; normal/regular priority
14071                              <1>     ; 21/05/2016
14072                              <1>     ;cmp    byte [u.pri], 2       ; high priority (run for event) ?
14073                              <1>     ;jnb    short swap
14074                              <1>     ; 16/01/2017
14075                              <1>     ; (Normal and also high/event priority processes will be added to
14076                              <1>     ; normal priority run queue for ensuring circular running sequence!)
14077                              <1>     ; (Timer interrupt or 'syspri' system call may change priority and run
14078                              <1>     ; queue to high/event level.)
14079 00010904 803D[7E8E0100]00    <1>     cmp     byte [u.pri], 0
14080 0001090B 7702                <1>     ja      short tswap_1  ; normal priority run queue
14081                              <1>     ;
14082 0001090D 43                  <1>     inc     ebx
14083 0001090E 43                  <1>     inc     ebx            ; runq+4, 'runq_background', low priority
14084                              <1> tswap_1:
14085 0001090F A0[858E0100]        <1>     mov     al, [u.uno]
14086                              <1>             ; movb u.uno,r1 / move users process number to r1
14087                              <1>             ; mov  $runq+4,r2
14088                              <1>                     ; / move lowest priority queue address to r2
14089                              <1>             ; ebx = run queue
14090 00010914 E8FD000000          <1>     call    putlu
14091                              <1>             ; jsr r0,putlu / create link from last user on Q to
14092                              <1>                         ; / u.uno's user
14093                              <1>
```

```
14094                           <1> switch: ; Retro UNIX 386 v1
14095                           <1> swap:
14096                           <1>     ; 20/08/2024
14097                           <1>     ; 02/01/2017
14098                           <1>     ; 21/05/2016
14099                           <1>     ; 20/05/2016
14100                           <1>     ; 02/05/2016
14101                           <1>     ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
14102                           <1>     ; 10/05/2015 - 02/09/2015 (Retro UNIX 386 v1)
14103                           <1>     ; 14/04/2013 - 08/03/2014 (Retro UNIX 8086 v1)
14104                           <1>     ;
14105                           <1>     ; 'swap' is routine that controls the swapping of processes
14106                           <1>     ; in and out of core.
14107                           <1>     ;
14108                           <1>     ; TRDOS 386 (TRDOS v2.0) modification ->  ** 20/05/2016 **
14109                           <1>     ;     * 3 different priority level is applied
14110                           <1>     ;       (just as original unix v1)
14111                           <1>     ;       1) high priority (event) run queue, 'runq_event'
14112                           <1>     ;       2) normal priority (regular) run queue, 'runq_normal'
14113                           <1>     ;       3) low priority (background) run queue, 'runq_backgroud'
14114                           <1>     ;       'swap' code will run a process which has max. priority
14115                           <1>       ;       (for earliest event at first)
14116                           <1>     ;
14117                           <1>     ; Retro UNIX 386 v1 modification ->
14118                           <1>     ;       swap (software task switch) is performed by changing
14119                           <1>     ;       user's page directory (u.pgdir) instead of segment change
14120                           <1>     ;       as in Retro UNIX 8086 v1.
14121                           <1>     ;
14122                           <1>     ; RETRO UNIX 8086 v1 modification ->
14123                           <1>     ;       'swap to disk' is replaced with 'change running segment'
14124                           <1>     ;       according to 8086 cpu (x86 real mode) architecture.
14125                           <1>     ;       pdp-11 was using 64KB uniform memory while IBM PC
14126                           <1>     ;       compatibles was using 1MB segmented memory
14127                           <1>     ;       in 8086/8088 times.
14128                           <1>     ;
14129                           <1>     ; INPUTS ->
14130                           <1>     ;    runq table - contains processes to run.
14131                           <1>     ;    p.link - contains next process in line to be run.
14132                           <1>     ;    u.uno - process number of process in core
14133                           <1>     ;    s.stack - swap stack used as an internal stack for swapping.
14134                           <1>     ; OUTPUTS ->
14135                           <1>     ;    (original unix v1 -> present process to its disk block)
14136                           <1>     ;    (original unix v1 -> new process into core ->
14137                           <1>     ;         Retro Unix 8086 v1 -> segment registers changed
14138                           <1>     ;         for new process)
14139                           <1>     ;    u.quant = 3 (Time quantum for a process)
14140                           <1>     ;       ((INT 1Ch count down speed -> 18.2 times per second)
14141                           <1>     ;    RETRO UNIX 8086 v1 will use INT 1Ch (18.2 times per second)
14142                           <1>     ;       for now, it will swap the process if there is not
14143                           <1>     ;       a keyboard event (keystroke) (Int 15h, function 4Fh)
14144                           <1>     ;       or will count down from 3 to 0 even if there is a
14145                           <1>     ;        keyboard event locking due to repetitive key strokes.
14146                           <1>     ;       u.quant will be reset to 3 for RETRO UNIX 8086 v1.
14147                           <1>     ;
14148                           <1>     ; ((Modified registers: EAX, EDX, EBX, ECX, ESI, EDI))
14149                           <1>
14150                           <1>     ;NOTE:
14151                           <1>     ;High priority queue is the first for selecting a process to run.
14152                           <1>     ;If there is not a process in high priority level run queue,
14153                           <1>     ;a process in normal priority run queue will be selected
14154                           <1>     ;or a proces in low priority run queue will be selected if normal
14155                           <1>     ;priority level run queue is empty.
14156                           <1>
14157                           <1>     ; 21/05/2016 -(3 priority levels, 3 run queues)
14158 00010919 BE[228E0100]     <1>     mov     esi, runq ; 'runq_event' ; high priority, 'run for event'
14159 0001091E C605[10830100]03 <1>     mov     byte [priority], 3 ; high priority + 1
14160 00010925 31DB             <1>     xor     ebx, ebx ; 02/01/2017
14161                           <1> swap_0: ; 1: / search runq table for highest priority process
14162 00010927 66AD             <1>     lodsw  ; mov ax, [esi], add esi+2
14163                           <1>     ;xor    ebx, ebx ; 02/05/2016
14164 00010929 6621C0           <1>     and     ax, ax ; are there any processes to run in this Q entry
14165 0001092C 750E             <1>     jnz     short swap_2
14166                           <1>     ; 21/05/2026
14167                           <1>     ; runq_normal = runq+2, runq_background = runq+4
14168 0001092E FE0D[10830100]   <1>     dec     byte [priority] ; 3 -> 3, 2 -> 1, 1-> 0
14169 00010934 75F1             <1>     jnz     short swap_0
14170                           <1>     ;cmp    esi, runq+6  ; if zero compare address to end of table
14171                           <1>     ;jb     short swap_0 ; if not at end, go back
14172                           <1> swap_1:
14173                           <1>     ; 02/05/2016
14174                           <1>     ; 29/04/2016 (TRDOS 386 = TRDOS v2.0)
14175                           <1>     ; No user process to run...
14176                           <1>     ; Run the kernel process... MainProg: Internal Command Interpreter
14177 00010936 FEC0             <1>     inc     al ; mov al, 1  ; process number of MainProg
14178 00010938 FEC3             <1>     inc     bl ; mov bl, al ; 1
14179 0001093A EB1D             <1>     jmp     short swap_4
14180                           <1> swap_2:
14181                           <1>     ; 21/05/2016
14182 0001093C FE0D[10830100]   <1>     dec     byte [priority] ; priority level of present user/process
14183                           <1>                             ; 0, 1, 2
14184 00010942 4E               <1>     dec     esi
14185 00010943 4E               <1>         dec     esi
14186                           <1>     ;
14187 00010944 88C3             <1>     mov     bl, al
14188 00010946 38E0             <1>     cmp     al, ah ; is there only 1 process in the queue to be run
14189 00010948 740A             <1>     je      short swap_3 ; yes
14190 0001094A 8AA3[578D0100]   <1>     mov     ah, [ebx+p.link-1]
14191 00010950 8826             <1>             mov     [esi], ah ; move next process in line into run queue
14192 00010952 EB05             <1>     jmp     short swap_4
14193                           <1> swap_3:
14194                           <1>     ;xor    dx, dx
14195                           <1>     ; 20/08/2024
14196 00010954 31D2             <1>     xor     edx, edx
14197 00010956 668916           <1>     mov     [esi], dx ; zero the entry; no processes on the Q
14198                           <1> swap_4:
14199 00010959 8A25[858E0100]   <1>     mov     ah, [u.uno]
14200 0001095F 38C4             <1>     cmp     ah, al ; is this process the same as the process in core?
14201 00010961 743B             <1>             je      short swap_8 ; yes, don't have to swap
14202 00010963 08E4             <1>     or      ah, ah ; is the process # = 0
14203 00010965 740D             <1>             jz      short swap_6 ; 'sysexit'
14204                           <1>     ;cmp    ah, al ; is this process the same as the process in core?
14205                           <1>             ;je     short swap_8 ; yes, don't have to swap
14206 00010967 8925[308E0100]   <1>     mov     [u.usp], esp ; return address for 'syswait' & 'sleep'
14207 0001096D E834000000       <1>     call    wswap  ; write out core to disk
14208 00010972 EB1C             <1>     jmp     short swap_7
14209                           <1> swap_6:
14210                           <1>     ; Deallocate memory pages belong to the process
14211                           <1>     ; which is being terminated.
14212                           <1>     ; (Retro UNIX 386 v1 modification !)
14213                           <1>     ;
14214 00010974 53               <1>     push    ebx
14215 00010975 A1[8C8E0100]     <1>     mov     eax, [u.pgdir]  ; page directory of the process
14216 0001097A 8B1D[908E0100]   <1>     mov     ebx, [u.ppgdir] ; page directory of the parent process
14217 00010980 E8D14FFFFF       <1>     call    deallocate_page_dir
```

```
14218 00010985 A1[888E0100]        <1>     mov     eax, [u.upage] ; 'user' structure page of the process
14219 0001098A E85650FFFF          <1>     call    deallocate_page
14220 0001098F 5B                  <1>     pop     ebx
14221                              <1> swap_7:
14222 00010990 C0E302              <1>     shl     bl, 2 ; * 4
14223                              <1>     ;;;
14224 00010993 8B83[748D0100]      <1>     mov     eax, [ebx+p.upage-4] ; the 'u' page of the new process
14225 00010999 E840000000          <1>     call    rswap ; read new process into core
14226                              <1> swap_8:
14227                              <1>     ; Retro UNIX  8086 v1 modification !
14228 0001099E C605[7C8E0100]04    <1>     mov     byte [u.quant], time_count
14229 000109A5 C3                  <1>     retn
14230                              <1>
14231                              <1> wswap:  ; < swap out, swap to disk >
14232                              <1>     ; 28/02/2017 (fnsave)
14233                              <1>     ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
14234                              <1>     ; 09/05/2015 (Retro UNIX 386 v1)
14235                              <1>     ; 26/05/2013 - 08/03/2014 (Retro UNIX 8086 v1)
14236                              <1>     ; 'wswap' writes out the process that is in core onto its
14237                              <1>     ; appropriate disk area.
14238                              <1>     ;
14239                              <1>     ; Retro UNIX 386 v1 modification ->
14240                              <1>     ;       User (u) structure content and the user's register content
14241                              <1>     ;      will be copied to the process's/user's UPAGE (a page for
14242                              <1>     ;      saving 'u' structure and user registers for task switching).
14243                              <1>     ;      u.usp - points to kernel stack address which contains
14244                              <1>     ;              user's registers while entering system call.
14245                              <1>     ;      u.sp  - points to kernel stack address
14246                              <1>     ;              to return from system call -for IRET-.
14247                              <1>     ;      [u.usp]+32+16 = [u.sp]
14248                              <1>     ;      [u.usp] -> edi, esi, ebp, esp (= [u.usp]+32), ebx,
14249                              <1>     ;              edx, ecx, eax, gs, fs, es, ds, -> [u.sp].
14250                              <1>     ;
14251                              <1>     ; Retro UNIX 8086 v1 modification ->
14252                              <1>     ;       'swap to disk' is replaced with 'change running segment'
14253                              <1>     ;      according to 8086 cpu (x86 real mode) architecture.
14254                              <1>     ;      pdp-11 was using 64KB uniform memory while IBM PC
14255                              <1>     ;      compatibles was using 1MB segmented memory
14256                              <1>     ;      in 8086/8088 times.
14257                              <1>     ;
14258                              <1>     ; INPUTS ->
14259                              <1>     ;    u.break - points to end of program
14260                              <1>     ;    u.usp - stack pointer at the moment of swap
14261                              <1>     ;    core - beginning of process program
14262                              <1>     ;    ecore - end of core
14263                              <1>     ;    user - start of user parameter area
14264                              <1>     ;    u.uno - user process number
14265                              <1>     ;    p.dska - holds block number of process
14266                              <1>     ; OUTPUTS ->
14267                              <1>     ;    swp I/O queue
14268                              <1>     ;    p.break - negative word count of process
14269                              <1>     ;    r1 - process disk address
14270                              <1>     ;    r2 - negative word count
14271                              <1>     ;
14272                              <1>     ; RETRO UNIX 8086 v1 input/output:
14273                              <1>     ;
14274                              <1>     ; INPUTS ->
14275                              <1>     ;    u.uno - process number (to be swapped out)
14276                              <1>     ; OUTPUTS ->
14277                              <1>     ;    none
14278                              <1>     ;
14279                              <1>     ;   ((Modified registers: ECX, ESI, EDI))
14280                              <1>     ;
14281                              <1>
14282                              <1>     ; 28/02/2017
14283                              <1>     ;cmp    byte [multi_tasking], 0  ; Musti tasking mode ?
14284                              <1>     ;jna    short wswp
14285 000109A6 803D[B38E0100]00    <1>     cmp     byte [u.fpsave], 0 ; 28/02/2017
14286 000109AD 7606                <1>     jna     short wswp
14287 000109AF DD35[B48E0100]      <1>     fnsave  [u.fpregs] ; save floating point registers (94 bytes)
14288                              <1> wswp:
14289 000109B5 8B3D[888E0100]      <1>     mov     edi, [u.upage] ; process's user (u) structure page addr
14290 000109BB B93D000000          <1>     mov     ecx, (U_SIZE + 3) / 4
14291 000109C0 BE[2C8E0100]        <1>     mov     esi, user ; active user (u) structure
14292 000109C5 F3A5                <1>     rep     movsd
14293                              <1>     ;
14294 000109C7 8B35[308E0100]      <1>     mov     esi, [u.usp] ; esp (system stack pointer,
14295                              <1>                          ;      points to user registers)
14296 000109CD 8B0D[2C8E0100]      <1>     mov     ecx, [u.sp]  ; return address from the system call
14297                              <1>                          ; (for IRET)
14298                              <1>                          ; [u.sp] -> EIP (user)
14299                              <1>                          ; [u.sp+4]-> CS (user)
14300                              <1>                          ; [u.sp+8] -> EFLAGS (user)
14301                              <1>                          ; [u.sp+12] -> ESP (user)
14302                              <1>                          ; [u.sp+16] -> SS (user)
14303 000109D3 29F1                <1>     sub     ecx, esi     ; required space for user registers
14304 000109D5 83C114              <1>     add     ecx, 20      ; +5 dwords to return from system call
14305                              <1>                          ; (for IRET)
14306 000109D8 C1E902              <1>     shr     ecx, 2
14307 000109DB F3A5                <1>     rep     movsd
14308 000109DD C3                  <1>     retn
14309                              <1>
14310                              <1> rswap:  ; < swap in, swap from disk >
14311                              <1>     ; 28/02/2017 (frstor)
14312                              <1>     ; 15/01/2017
14313                              <1>     ; 14/01/2017
14314                              <1>     ; 21/05/2016
14315                              <1>     ; 03/05/2016
14316                              <1>     ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
14317                              <1>     ; 09/05/2015 - 15/09/2015 (Retro UNIX 386 v1)
14318                              <1>     ; 26/05/2013 - 08/03/2014 (Retro UNIX 8086 v1)
14319                              <1>     ; 'rswap' reads a process whose number is in r1,
14320                              <1>     ; from disk into core.
14321                              <1>     ;
14322                              <1>     ; Retro UNIX 386 v1 modification ->
14323                              <1>     ;       User (u) structure content and the user's register content
14324                              <1>     ;      will be restored from process's/user's UPAGE (a page for
14325                              <1>     ;      saving 'u' structure and user registers for task switching).
14326                              <1>     ;      u.usp - points to kernel stack address which contains
14327                              <1>     ;              user's registers while entering system call.
14328                              <1>     ;      u.sp  - points to kernel stack address
14329                              <1>     ;              to return from system call -for IRET-.
14330                              <1>     ;      [u.usp]+32+16 = [u.sp]
14331                              <1>     ;      [u.usp] -> edi, esi, ebp, esp (= [u.usp]+32), ebx,
14332                              <1>     ;              edx, ecx, eax, gs, fs, es, ds, -> [u.sp].
14333                              <1>     ;
14334                              <1>     ; RETRO UNIX 8086 v1 modification ->
14335                              <1>     ;       'swap to disk' is replaced with 'change running segment'
14336                              <1>     ;      according to 8086 cpu (x86 real mode) architecture.
14337                              <1>     ;      pdp-11 was using 64KB uniform memory while IBM PC
14338                              <1>     ;      compatibles was using 1MB segmented memory
14339                              <1>     ;      in 8086/8088 times.
14340                              <1>     ;
14341                              <1>     ; INPUTS ->
```

```
14342                                   <1>     ;    r1 - process number of process to be read in
14343                                   <1>     ;    p.break - negative of word count of process
14344                                   <1>     ;    p.dska - disk address of the process
14345                                   <1>     ;    u.emt - determines handling of emt's
14346                                   <1>     ;    u.ilgins - determines handling of illegal instructions
14347                                   <1>     ; OUTPUTS ->
14348                                   <1>     ;    8 = (u.ilgins)
14349                                   <1>     ;    24 = (u.emt)
14350                                   <1>     ;    swp - bit 10 is set to indicate read
14351                                   <1>     ;              (bit 15=0 when reading is done)
14352                                   <1>     ;    swp+2 - disk block address
14353                                   <1>     ;    swp+4 - negative word count
14354                                   <1>     ;      ((swp+6 - address of user structure))
14355                                   <1>     ;
14356                                   <1>     ; RETRO UNIX 8086 v1 input/output:
14357                                   <1>     ;
14358                                   <1>     ; INPUTS ->
14359                                   <1>     ;    AL - new process number (to be swapped in)
14360                                   <1>     ; OUTPUTS ->
14361                                   <1>     ;    none
14362                                   <1>     ;
14363                                   <1>     ;   ((Modified registers: EAX, ECX, ESI, EDI, ESP))
14364                                   <1>     ;
14365                                   <1>     ; Retro UNIX 386 v1 - modification ! 14/05/2015
14366 000109DE 89C6                    <1>     mov     esi, eax  ; process's user (u) structure page addr
14367 000109E0 B93D000000              <1>     mov     ecx, (U_SIZE + 3) / 4
14368 000109E5 BF[2C8E0100]            <1>     mov     edi, user ; active user (u) structure
14369 000109EA F3A5                    <1>     rep     movsd
14370 000109EC 58                      <1>     pop     eax     ; 'rswap' return address
14371                                   <1>     ;
14372                                   <1>     ;cli
14373 000109ED 8B3D[308E0100]          <1>     mov     edi, [u.usp] ; esp (system stack pointer,
14374                                   <1>                          ;     points to user registers)
14375 000109F3 89FC                    <1>     mov     esp, edi     ; 14/01/2017
14376 000109F5 8B0D[2C8E0100]          <1>     mov     ecx, [u.sp]  ; return address from the system call
14377                                   <1>                          ; (for IRET)
14378                                   <1>                          ; [u.sp] -> EIP (user)
14379                                   <1>                          ; [u.sp+4]-> CS (user)
14380                                   <1>                          ; [u.sp+8] -> EFLAGS (user)
14381                                   <1>                          ; [u.sp+12] -> ESP (user)
14382                                   <1>                          ; [u.sp+16] -> SS (user)
14383 000109FB 29F9                    <1>     sub     ecx, edi     ; required space for user registers
14384 000109FD 83C114                  <1>     add     ecx, 20      ; +5 dwords to return from system call
14385                                   <1>                          ; (for IRET)
14386 00010A00 C1E902                  <1>     shr     ecx, 2
14387 00010A03 F3A5                    <1>     rep     movsd
14388                                   <1>     ;mov    esp, [u.usp] ; 15/09/2015
14389                                   <1>     ;sti
14390                                   <1>     ; 28/02/2017
14391                                   <1>     ;cmp    byte [multi_tasking], 0  ; Multi tasking mode ?
14392                                   <1>     ;jna    short rswp_retn
14393 00010A05 803D[B38E0100]00        <1>     cmp     byte [u.fpsave], 0
14394 00010A0C 7606                    <1>     jna     short rswp_retn
14395 00010A0E DD25[B48E0100]          <1>     frstor  [u.fpregs] ; restore floating point regs (94 bytes)
14396                                   <1>                        ; 108 bytes (22/08/2024)
14397                                   <1> rswp_retn:
14398 00010A14 50                      <1>     push    eax     ; 'rswap' return address
14399 00010A15 C3                      <1>     retn
14400                                   <1>
14401                                   <1> putlu:
14402                                   <1>     ; 20/05/2016
14403                                   <1>     ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
14404                                   <1>     ; 10/05/2015 - 12/09/2015 (Retro UNIX 386 v1)
14405                                   <1>     ; 15/04/2013 - 23/02/2014 (Retro UNIX 8086 v1)
14406                                   <1>     ; 'putlu' is called with a process number in r1 and a pointer
14407                                   <1>     ; to lowest priority Q (runq+4) in r2. A link is created from
14408                                   <1>     ; the last process on the queue to process in r1 by putting
14409                                   <1>     ; the process number in r1 into the last process's link.
14410                                   <1>     ;
14411                                   <1>     ; INPUTS ->
14412                                   <1>     ;    r1 - user process number
14413                                   <1>     ;    r2 - points to lowest priority queue
14414                                   <1>     ;    p.dska - disk address of the process
14415                                   <1>     ;    u.emt - determines handling of emt's
14416                                   <1>     ;    u.ilgins - determines handling of illegal instructions
14417                                   <1>     ; OUTPUTS ->
14418                                   <1>     ;    r3 - process number of last process on the queue upon
14419                                   <1>     ;         entering putlu
14420                                   <1>     ;    p.link-1 + r3 - process number in r1
14421                                   <1>     ;    r2 - points to lowest priority queue
14422                                   <1>     ;
14423                                   <1>     ; ((Modified registers: EDX, EBX))
14424                                   <1>     ;
14425                                   <1>     ; / r1 = user process no.; r2 points to lowest priority queue
14426                                   <1>
14427                                   <1>     ; EBX = r2
14428                                   <1>     ; EAX = r1 (AL=r1b)
14429                                   <1>
14430                                   <1>     ; 20/05/2016
14431                                   <1>     ; AL = process number (1 to 16) // Retro UNIX 8086, 386 v1 //
14432                                   <1>     ;     (max. 16 processes available for current kernel version)
14433                                   <1>     ; EBX = run queue address ; 20/05/2016 (TRDOS 386)
14434                                   <1>             ; which is one of following addresses:
14435                                   <1>             ;  1) 'runq_event' high priority run queue
14436                                   <1>             ;  2) 'runq_normal' normal/regular priority run queue
14437                                   <1>             ;  3) 'runq_background' low priority run queue
14438                                   <1>
14439                                   <1>     ;mov    ebx, runq
14440 00010A16 0FB613                  <1>     movzx   edx, byte [ebx]
14441 00010A19 43                      <1>     inc     ebx
14442 00010A1A 20D2                    <1>     and     dl, dl
14443                                   <1>             ; tstb (r2)+ / is queue empty?
14444 00010A1C 740A                    <1>             jz      short putlu_1
14445                                   <1>             ; beq 1f / yes, branch
14446 00010A1E 8A13                    <1>     mov     dl, [ebx] ; 12/09/2015
14447                                   <1>             ; movb (r2),r3 / no, save the "last user" process number
14448                                   <1>                      ; / in r3
14449 00010A20 8882[578D0100]          <1>             mov     [edx+p.link-1], al
14450                                   <1>             ; movb r1,p.link-1(r3) / put pointer to user on
14451                                   <1>                      ; / "last users" link
14452 00010A26 EB03                    <1>     jmp     short putlu_2
14453                                   <1>             ; br 2f /
14454                                   <1> putlu_1: ; 1:
14455 00010A28 8843FF                  <1>     mov     [ebx-1], al
14456                                   <1>                     ; movb r1,-1(r2) / user is only user;
14457                                   <1>                        ; / put process no. at beginning and at end
14458                                   <1> putlu_2: ; 2:
14459 00010A2B 8803                    <1>     mov     [ebx], al
14460                                   <1>                     ; movb r1,(r2) / user process in r1 is now the last entry
14461                                   <1>                         ; / on the queue
14462 00010A2D 88C2                    <1>     mov     dl, al
14463 00010A2F 88B2[578D0100]          <1>         mov     [edx+p.link-1], dh ; 0
14464                                   <1>             ; dec r2 / restore r2
14465 00010A35 C3                      <1>         retn
```

```
14466                                 <1>             ; rts r0
14467                                 <1>
14468                                 <1> sysver:
14469                                 <1>     ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
14470 00010A36 C705[348E0100]0002-    <1>     mov     dword [u.r0], 200h ; AH = major version, AL = minor version
14470 00010A3E 0000                   <1>
14471 00010A40 E939C0FFFF             <1>     jmp     sysret
14472                                 <1>
14473                                 <1> syspri: ; change running priority (of the process)
14474                                 <1>     ; 23/07/2022 - TRDOS 386 v2.0.5
14475                                 <1>     ; 21/05/2016
14476                                 <1>     ; 20/05/2026 - TRDOS 386 (TRDOS v2.0)
14477                                 <1>     ; INPUT ->
14478                                 <1>     ;       BL = priority level
14479                                 <1>     ;          0 = low running priority (running on background)
14480                                 <1>     ;          1 = normal/regular priority (running as regular)
14481                                 <1>     ;          2 = high/event priority (running for event)
14482                                 <1>     ;          >2 = invalid, it will accepted as 2 (event)
14483                                 <1>     ;          0FFh = get/return current running priority only
14484                                 <1>     ; OUTPUT ->
14485                                 <1>     ;       * if current [u.pri] < 2
14486                                 <1>     ;         if BL input < 0FFh ->
14487                                 <1>     ;            [u.pri] is updated as in BL input (0,1,2)
14488                                 <1>     ;         if BL input = 0FFh -> AL = [u.pri] (current)
14489                                 <1>     ;
14490                                 <1>     ;       * if current [u.pri] = 2
14491                                 <1>     ;         if BL input < 0FFh -> cf = 1 & AL = 2
14492                                 <1>     ;         if BL input = 0FFh -> cf = 0 & AL = 2
14493                                 <1>     ;
14494                                 <1>     ;       NOTE:
14495                                 <1>     ;       If [u.pri] = 2, it can not be changed to 1 or 0;
14496                                 <1>     ;       because, run queue of the running process is unspecified
14497                                 <1>     ;       at this stage. Process might be started by a timer event
14498                                 <1>     ;       or priority might be changed to high by previous
14499                                 <1>     ;       'syspri' system       call. In both cases, the process is in
14500                                 <1>     ;       'runq_normal' or 'runq_background' queue.
14501                                 <1>     ;       As result of this fact, when the [u.quant] time quantum
14502                                 <1>     ;       of the process is elapsed or 'sysrele' system call is
14503                                 <1>     ;       instructed by the process, 'tswap' ('tswitch') procedure
14504                                 <1>     ;       will be called (to 'swap' or 'switch' out the procedure)
14505                                 <1>     ;       and it will not call 'putlu' to add the (stopping)
14506                                 <1>     ;       process to relevant run queue when [u.pri] = 2.
14507                                 <1>     ;       (Otherwise, it would be possible to add process to
14508                                 <1>     ;       a run queue while it is already in a run queue, wrongly.)
14509                                 <1>     ;
14510                                 <1>     ;       If [u.pri]< 2, 'tswap/tswitch' procedure will call
14511                                 <1>     ;       'putlu' to add process to relevant run queue
14512                                 <1>     ;       according to [u.pri] value. ('runq_normal' for 1,
14513                                 <1>     ;       'runq_background' for 0).
14514                                 <1>     ;
14515                                 <1>     ;       If BL input >= 2 and < 0FFh while [u.pri] < 2,
14516                                 <1>     ;       process will be added to 'runq_normal' queue and
14517                                 <1>     ;       [u.pri] will be set to 2. (in 'syspri' system call)
14518                                 <1>     ;
14519                                 <1>
14520 00010A45 29C0                   <1>     sub     eax, eax ; 0
14521 00010A47 A3[A08E0100]           <1>     mov     [u.error], eax
14522                                 <1>
14523 00010A4C A0[7E8E0100]           <1>     mov     al, [u.pri]
14524 00010A51 A3[348E0100]           <1>     mov     [u.r0], eax
14525                                 <1>
14526 00010A56 FEC3                   <1>     inc     bl
14527                                 <1>     ;jz     sysret ; 0FFh -> 0, get priority level
14528                                 <1>     ; 23/07/2022
14529 00010A58 742C                   <1>     jz      short syspri_2 ; jmp sysret
14530                                 <1>
14531 00010A5A 3C02                   <1>     cmp     al, 2
14532                                 <1>     ;jnb    error ; CF = 1 & AL = 2 (& last error = 0)
14533                                 <1>     ; 23/07/2022
14534 00010A5C 7205                   <1>     jb      short syspri_0
14535 00010A5E E9FBBFFFFF             <1>     jmp     error
14536                                 <1> syspri_0:
14537 00010A63 FECB                   <1>     dec     bl
14538 00010A65 80FB02                 <1>     cmp     bl, 2
14539 00010A68 7602                   <1>     jna     short syspri_1
14540 00010A6A B302                   <1>     mov     bl, 2
14541                                 <1> syspri_1:
14542 00010A6C 881D[7E8E0100]         <1>     mov     [u.pri], bl
14543 00010A72 80FB02                 <1>     cmp     bl, 2
14544                                 <1>     ;jb     sysret
14545                                 <1>     ; 23/07/2022
14546 00010A75 720F                   <1>     jb      short syspri_2 ; jmp sysret
14547                                 <1>
14548                                 <1>     ; here...
14549                                 <1>     ; Priority of current process has been changed to high
14550                                 <1>     ; ('run for event') but current process will be added to
14551                                 <1>     ; 'run as normal' queue. ('run for event' high priority
14552                                 <1>     ; queue is under control of timer -& RTC- interrupt only!)
14553                                 <1>     ;
14554                                 <1>     ; (Otherwise, process can fall into black hole!
14555                                 <1>     ; e.g. if it is not in waiting list and it has not got
14556                                 <1>     ; a timer event and it is not in a run queue!
14557                                 <1>     ; Because, when [u.pri] is 2, 'tswap/tswitch' will not
14558                                 <1>     ; add the stopping process to a run queue.)
14559                                 <1>
14560 00010A77 A0[858E0100]           <1>     mov     al, [u.uno]
14561 00010A7C BB[248E0100]           <1>     mov     ebx, runq_normal ; normal priority !
14562                                 <1>                              ; [u.pri] is set to high
14563                                 <1>                              ; but 'runq_event' queue is set
14564                                 <1>                              ; only by the kernel's timer
14565                                 <1>                              ; event function (timer interrupt).
14566 00010A81 E890FFFFFF             <1>     call    putlu
14567                                 <1> syspri_2:
14568 00010A86 E9F3BFFFFF             <1>     jmp     sysret
14569                                 <1>
14570                                 <1> cpass: ; / get next character from user area of core and put it in AL (r1)
14571                                 <1>     ; 30/07/2022 - TRDOS 386 Kernel v2.0.5
14572                                 <1>     ; 02/05/2016 - TRDOS 386 (TRDOS v2.0)
14573                                 <1>     ; 19/05/2015 - 18/10/2015 (Retro UNIX 386 v1)
14574                                 <1>     ; 14/08/2013 - 20/09/2013 (Retro UNIX 8086 v1)
14575                                 <1>     ; INPUTS ->
14576                                 <1>     ;     [u.base] = virtual address in user area
14577                                 <1>     ;     [u.count] = byte count (max.)
14578                                 <1>     ;     [u.pcount] = byte count in page (0 = reset)
14579                                 <1>     ; OUTPUTS ->
14580                                 <1>     ;     AL = the character which is pointed by [u.base]
14581                                 <1>     ;     zf = 1 -> transfer count has been completed
14582                                 <1>       ;
14583                                 <1>     ; ((Modified registers: EAX, EDX, ECX))
14584                                 <1>
14585                                 <1>     ; 30/07/2022
14586 00010A8B 29C0                   <1>     sub     eax, eax
14587                                 <1>
14588 00010A8D 3905[5C8E0100]         <1>     cmp     [u.count], eax ; 0
```

```
14589                                   <1>     ;cmp    dword [u.count], 0  ; have all the characters been transferred
14590                                   <1>                                 ; i.e., u.count, # of chars. left
14591 00010A93 763D                     <1>     jna     short cpass_3       ; to be transferred = 0?) yes, branch
14592 00010A95 FF0D[5C8E0100]           <1>     dec     dword [u.count]           ; no, decrement u.count
14593                                   <1>         ; 19/05/2015
14594                                   <1>     ;(Retro UNIX 386 v1 - translation from user's virtual address
14595                                   <1>     ;                     to physical address
14596                                   <1>     ; 30/07/2022
14597 00010A9B 663905[988E0100]         <1>     cmp     [u.pcount], ax ; 0
14598                                   <1>     ;cmp    word [u.pcount], 0 ; byte count in page = 0 (initial value)
14599                                   <1>                         ; 1-4095 --> use previous physical base address
14600                                   <1>                         ; in [u.pbase]
14601 00010AA2 770D                     <1>     ja      short cpass_1
14602                                   <1>     ; 30/07/2022
14603 00010AA4 3905[908E0100]           <1>     cmp     [u.ppgdir], eax ; 0
14604                                   <1>     ;cmp    dword [u.ppgdir], 0 ; is the caller os kernel
14605 00010AAA 7427                     <1>         je      short cpass_k       ; (sysexec, '/etc/init') ?  (MainProg)
14606 00010AAC E861FDFFFF               <1>     call    trans_addr_r
14607                                   <1> cpass_1:
14608 00010AB1 66FF0D[988E0100]         <1>     dec     word [u.pcount]
14609                                   <1> cpass_2:
14610 00010AB8 8B15[948E0100]           <1>     mov     edx, [u.pbase]
14611 00010ABE 8A02                     <1>     mov     al, [edx]       ; take the character pointed to
14612                                   <1>                             ; by u.base and put it in r1
14613 00010AC0 FF05[608E0100]           <1>     inc     dword [u.nread] ; increment no. of bytes transferred
14614 00010AC6 FF05[588E0100]           <1>     inc     dword [u.base]  ; increment the buffer address to point to the
14615                                   <1>                             ; next byte
14616 00010ACC FF05[948E0100]           <1>     inc     dword [u.pbase]
14617                                   <1> cpass_3:
14618 00010AD2 C3                       <1>     retn
14619                                   <1> cpass_k:
14620                                   <1>     ; 02/07/2015
14621                                   <1>     ; The caller is os kernel
14622                                   <1>     ; (get sysexec arguments from kernel's memory space)
14623 00010AD3 8B1D[588E0100]           <1>     mov     ebx, [u.base]
14624 00010AD9 66C705[988E0100]00-      <1>         mov     word [u.pcount], PAGE_SIZE ; 4096
14624 00010AE1 10                       <1>
14625 00010AE2 891D[948E0100]           <1>     mov     [u.pbase], ebx
14626 00010AE8 EBCE                     <1>     jmp     short cpass_2
14627                                   <1>
14628                                   <1> transfer_to_user_buffer: ; fast transfer
14629                                   <1>     ; 27/05/2016
14630                                   <1>     ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
14631                                   <1>     ;
14632                                   <1>     ; INPUT ->
14633                                   <1>     ;       ESI = source address in system space
14634                                   <1>     ;       EDI = user's buffer address
14635                                   <1>     ;       ECX = transfer (byte) count
14636                                   <1>     ;       [u.pgdir] = user's page directory
14637                                   <1>     ; OUTPUT ->
14638                                   <1>     ;       ECX = actual transfer count
14639                                   <1>     ;       cf = 1 -> error
14640                                   <1>     ;       [u.count] = remain byte count
14641                                   <1>     ;
14642                                   <1>     ; Modified registers: eax, ecx
14643                                   <1>     ;
14644                                   <1>
14645 00010AEA 21C9                     <1>     and     ecx, ecx
14646 00010AEC 743B                     <1>     jz      short ttub_4
14647                                   <1>
14648 00010AEE 890D[5C8E0100]           <1>     mov     [u.count], ecx
14649                                   <1>
14650 00010AF4 57                       <1>     push    edi
14651 00010AF5 56                       <1>     push    esi
14652 00010AF6 53                       <1>     push    ebx
14653 00010AF7 52                       <1>     push    edx
14654 00010AF8 51                       <1>     push    ecx
14655                                   <1>
14656 00010AF9 89FB                     <1>     mov     ebx, edi
14657 00010AFB 81C300004000             <1>     add     ebx, CORE ; 27/05/2016
14658                                   <1> ttub_1:
14659                                   <1>     ; ebx = virtual (linear) address
14660                                   <1>     ; [u.pgdir] = user's page directory
14661 00010B01 E80451FFFF               <1>             call    get_physical_addr_x ; get physical address
14662 00010B06 7222                     <1>     jc      short ttub_5
14663                                   <1>     ; eax = physical address
14664                                   <1>     ; ecx = remain byte count in page (1-4096)
14665 00010B08 89C7                     <1>     mov     edi, eax
14666 00010B0A A1[5C8E0100]             <1>     mov     eax, [u.count]
14667 00010B0F 39C1                     <1>     cmp     ecx, eax
14668 00010B11 7602                     <1>     jna     short ttub_2
14669 00010B13 89C1                     <1>     mov     ecx, eax
14670                                   <1> ttub_2:
14671 00010B15 29C8                     <1>     sub     eax, ecx
14672 00010B17 01CB                     <1>     add     ebx, ecx
14673 00010B19 F3A4                     <1>     rep     movsb
14674 00010B1B A3[5C8E0100]             <1>     mov     [u.count], eax
14675 00010B20 09C0                     <1>     or      eax, eax
14676 00010B22 75DD                     <1>     jnz     short ttub_1
14677                                   <1> ttub_retn:
14678                                   <1> tfub_retn:
14679 00010B24 59                       <1>     pop     ecx ; transfer count = actual transfer count
14680                                   <1> ttub_3:
14681 00010B25 5A                       <1>     pop     edx
14682 00010B26 5B                       <1>     pop     ebx
14683 00010B27 5E                       <1>     pop     esi
14684 00010B28 5F                       <1>     pop     edi
14685                                   <1> ttub_4:
14686 00010B29 C3                       <1>     retn
14687                                   <1> ttub_5:
14688 00010B2A 59                       <1>     pop     ecx
14689 00010B2B 2B0D[5C8E0100]           <1>     sub     ecx, [u.count] ; actual transfer count
14690 00010B31 F9                       <1>     stc
14691 00010B32 EBF1                     <1>     jmp     short ttub_3
14692                                   <1>
14693                                   <1> transfer_from_user_buffer: ; fast transfer
14694                                   <1>     ; 27/05/2016
14695                                   <1>     ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
14696                                   <1>     ;
14697                                   <1>     ; INPUT ->
14698                                   <1>     ;       ESI = user's buffer address
14699                                   <1>     ;       EDI = destination address in system space
14700                                   <1>     ;       ECX = transfer (byte) count
14701                                   <1>     ;       [u.pgdir] = user's page directory
14702                                   <1>     ; OUTPUT ->
14703                                   <1>     ;       ecx = actual transfer count
14704                                   <1>     ;       cf = 1 -> error
14705                                   <1>     ;       [u.count] = remain byte count
14706                                   <1>     ;
14707                                   <1>     ; Modified registers: eax, ecx
14708                                   <1>     ;
14709                                   <1>
14710 00010B34 21C9                     <1>     and     ecx, ecx
14711                                   <1>     ;jz     short tfub_4
```

```
14712 00010B36 74F1                <1>     jz      short ttub_4
14713                              <1>
14714 00010B38 890D[5C8E0100]      <1>     mov     [u.count], ecx
14715                              <1>
14716 00010B3E 57                  <1>     push    edi
14717 00010B3F 56                  <1>     push    esi
14718 00010B40 53                  <1>     push    ebx
14719 00010B41 52                  <1>     push    edx
14720 00010B42 51                  <1>     push    ecx
14721                              <1>
14722 00010B43 89F3                <1>     mov     ebx, esi
14723 00010B45 81C300004000        <1>     add     ebx, CORE ; 27/05/2016
14724                              <1> tfub_1:
14725                              <1>     ; ebx = virtual (linear) address
14726                              <1>     ; [u.pgdir] = user's page directory
14727 00010B4B E8BA50FFFF          <1>             call    get_physical_addr_x ; get physical address
14728                              <1>     ;jc     short tfub_5
14729 00010B50 72D8                <1>     jc      short ttub_5
14730                              <1>     ; eax = physical address
14731                              <1>     ; ecx = remain byte count in page (1-4096)
14732 00010B52 89C6                <1>     mov     esi, eax
14733 00010B54 A1[5C8E0100]        <1>     mov     eax, [u.count]
14734 00010B59 39C1                <1>     cmp     ecx, eax
14735 00010B5B 7602                <1>     jna     short tfub_2
14736 00010B5D 89C1                <1>     mov     ecx, eax
14737                              <1> tfub_2:
14738 00010B5F 29C8                <1>     sub     eax, ecx
14739 00010B61 01CB                <1>     add     ebx, ecx
14740 00010B63 F3A4                <1>     rep     movsb
14741 00010B65 A3[5C8E0100]        <1>     mov     [u.count], eax
14742 00010B6A 09C0                <1>     or      eax, eax
14743 00010B6C 75DD                <1>     jnz     short tfub_1
14744                              <1>
14745 00010B6E EBB4                <1>     jmp     short tfub_retn
14746                              <1>
14747                              <1> ;tfub_retn:
14748                              <1> ;   pop     ecx ; transfer count = actual transfer count
14749                              <1> ;tfub_3:
14750                              <1> ;   pop     edx
14751                              <1> ;   pop     ebx
14752                              <1> ;   pop     esi
14753                              <1> ;   pop     edi
14754                              <1> ;tfub_4:
14755                              <1> ;   retn
14756                              <1> ;tfub_5:
14757                              <1> ;   pop     ecx
14758                              <1> ;   sub     ecx, [u.count] ; actual transfer count
14759                              <1> ;   stc
14760                              <1> ;   jmp     short tfub_3
14761                              <1>
14762                              <1> sysfff: ; <Find First File>
14763                              <1>     ; 25/08/2024 (TRDOS 386 Kernel v2.0.9)
14764                              <1>     ; 08/08/2022
14765                              <1>     ; 30/07/2022 (TRDOS 386 Kernel v2.0.5)
14766                              <1>     ; 17/10/2016
14767                              <1>     ; 16/10/2016
14768                              <1>     ; 15/10/2016 TRDOS 386 (TRDOS v2.0) feature only !
14769                              <1>     ;           -derived from TRDOS v1.0, INT_21H.ASM-
14770                              <1>     ;            ("loc_INT21h_find_first_file")
14771                              <1>     ; TRDOS 8086 (v1.0)
14772                              <1>         ;   07/08/2011
14773                              <1>         ;   Find First File
14774                              <1>     ;       INPUT:
14775                              <1>         ;       CX= Attributes
14776                              <1>         ;       DS:DX= Pointer to filename
14777                              <1>     ;       MSDOS OUTPUT:
14778                              <1>     ;           DTA: (Default address: PSP offset 80h)
14779                              <1>     ;           Offset  Descrription
14780                              <1>     ;           0       Reserved for use find next file
14781                              <1>     ;           21      Attribute of file found
14782                              <1>     ;           22      Time stamp of file
14783                              <1>     ;           24      Date stamp of file
14784                              <1>     ;           26      File size in bytes
14785                              <1>     ;           30      Filename and extension (zero terminated)
14786                              <1>     ;       If cf = 1:
14787                              <1>     ;           Error Codes: (in AX)
14788                              <1>     ;              2 - File not found
14789                              <1>     ;             18 - No more files
14790                              <1>         ;
14791                              <1>     ; TRDOS 386 (v2.0)
14792                              <1>     ; 15/10/2016
14793                              <1>     ;
14794                              <1>         ; INPUT ->
14795                              <1>         ;     CL = File attributes
14796                              <1>     ;            bit 0 (1) - Read only file (R)
14797                              <1>     ;             bit 1 (1) - Hidden file (H)
14798                              <1>         ;            bit 2 (1) - System file (R)
14799                              <1>     ;            bit 3 (1) - Volume label/name (V)
14800                              <1>         ;            bit 4 (1) - Subdirectory (D)
14801                              <1>     ;           bit 5 (1) - File has been archived (A)
14802                              <1>     ;         CH = 0 -> Return basic parameters (24 bytes)
14803                              <1>     ;         CH > 0 -> Return FindFile structure/table (128 bytes)
14804                              <1>         ;           EBX = Pointer to filename (ASCIIZ) -path-
14805                              <1>     ;         EDX = File parameters buffer address
14806                              <1>     ;             (buffer size = 24 bytes if CH input = 0)
14807                              <1>     ;             (buffer size = 128 bytes if CH input > 0)
14808                              <1>     ;
14809                              <1>     ; OUTPUT ->
14810                              <1>     ;         EAX = 0 if CH input > 0
14811                              <1>     ;         EAX = First cluster number of file if CH input = 0
14812                              <1>     ;         EDX = File parameters table/structure address
14813                              <1>     ;         Basic Parameters:
14814                              <1>     ;             Offset  Description
14815                              <1>     ;             ------  ---------------
14816                              <1>     ;             0       File Attributes
14817                              <1>     ;             1       Ambiguous filename chars are used sign
14818                              <1>     ;                     (0 = filename fits exactly with request)
14819                              <1>     ;                     (>0 = ambiguous filename chars are used)
14820                              <1>     ;             2       Time stamp of file
14821                              <1>     ;             4       Date stamp of file
14822                              <1>     ;             6       File size in bytes
14823                              <1>     ;             10      Short Filename (ASCIIZ, max. 13 bytes)
14824                              <1>     ;             23      Longname Length (1-255) if existing
14825                              <1>     ;
14826                              <1>     ;          cf = 1 -> Error code in AL
14827                              <1>     ;
14828                              <1>     ; Modified Registers: EAX (at the return of system call)
14829                              <1>     ;
14830                              <1>     ; TR-DOS FindFile (FFF) Structure (128 bytes):
14831                              <1>     ; 09/10/2011 (DIR.ASM) - 10/02/2016 (trdoskx.s)
14832                              <1>     ;
14833                              <1>     ; Offset       Parameter            Size
14834                              <1>     ; ------       ------------------   --------
14835                              <1>     ; 0            FindFile_Drv         1 byte
```

```
14836                                   <1>     ; 1           FindFile_Directory    65 bytes
14837                                   <1>     ; 66          FindFile_Name         13 bytes
14838                                   <1>     ; 79          FindFile_LongNameEntryLength 1 byte
14839                                   <1>     ;Above 80 bytes form
14840                                   <1>     ;TR-DOS Source/Destination File FullName Format/Structure
14841                                   <1>     ; 80          FindFile_AttributesMask 1 word
14842                                   <1>     ; 82          FindFile_DirEntry     32 bytes (*)
14843                                   <1>     ; 114         FindFile_DirFirstCluster 1 double word
14844                                   <1>     ; 118         FindFile_DirCluster   1 double word
14845                                   <1>     ; 122         FindFile_DirEntryNumber 1 word
14846                                   <1>     ; 124         FindFile_MatchCounter 1 word
14847                                   <1>     ; 126         FindFile_Reserved     1 word
14848                                   <1>     ; (*) MS-DOS, FAT 12-16-32 classic directory entry (32 bytes)
14849                                   <1>
14850                                   <1>     ;mov   [u.namep], ebx
14851                                   <1>     ; 16/10/2016
14852 00010B70 8915[34830100]          <1>     mov    [FFF_UBuffer], edx
14853 00010B76 66890D[39830100]        <1>     mov    [FFF_Attrib], cx ; [FFF_RType] = ch
14854                                   <1>                     ; Attributes in CL, return data type in CH
14855 00010B7D 89DE                    <1>     mov    esi, ebx
14856                                   <1>     ; file name is forced, change directory as temporary
14857                                   <1>     ;mov   ax, 1
14858                                   <1>     ;mov   [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
14859                                   <1>     ;call  set_working_path
14860 00010B7F E820110000              <1>     call   set_working_path_x ; 17/10/2016
14861 00010B84 731D                    <1>     jnc    short sysfff_0
14862                                   <1>
14863 00010B86 21C0                    <1>     and    eax, eax  ; 0 -> Bad Path!
14864 00010B88 7505                    <1>     jnz    short sysfff_err
14865                                   <1>
14866                                   <1>     ; eax = 0
14867 00010B8A B80C000000              <1>     mov    eax, ERR_DIR_NOT_FOUND ; Directory not found !
14868                                   <1> sysfff_err:
14869 00010B8F A3[348E0100]            <1>     mov    [u.r0], eax
14870 00010B94 A3[A08E0100]            <1>     mov    [u.error], eax
14871 00010B99 E8DB110000              <1>         call       reset_working_path
14872 00010B9E E9BBBEFFFF              <1>     jmp    error
14873                                   <1>
14874                                   <1> sysfff_0:
14875                                   <1>     ;;;
14876                                   <1>     ; 25/08/2024 (bugfix)
14877                                   <1>     ;mov   al, [esp] ; ???
14878 00010BA3 A0[39830100]            <1>     mov    al, [FFF_Attrib]
14879                                   <1>     ;;;
14880 00010BA8 08C0                    <1>     or     al, al
14881 00010BAA 7412                    <1>     jz     short sysfff_2
14882 00010BAC B410                    <1>     mov    ah, 10h
14883 00010BAE A808                    <1>     test   al, 08h
14884 00010BB0 7503                    <1>     jnz    short sysfff_1
14885 00010BB2 80CC08                  <1>     or     ah, 08h
14886                                   <1> sysfff_1:
14887 00010BB5 2410                    <1>     and    al, 10h ; Directory
14888 00010BB7 7405                    <1>     jz     short sysfff_2
14889 00010BB9 80E408                  <1>     and    ah, 08h
14890 00010BBC 30C0                    <1>     xor    al, al ; When a directory is searched,
14891                                   <1>                   ; filename will be returned even if
14892                                   <1>                   ; it is not a directory!
14893                                   <1>                   ; Because: (in order to prevent
14894                                   <1>                   ; creating a dir with existing file name)
14895                                   <1>                   ; Dir and file names must not be same!
14896                                   <1>                   ; (return attribute must be checked)
14897                                   <1> sysfff_2:
14898                                   <1>     ; AX = Attributes mask
14899                                   <1>             ; AL = AND mask (result must be equal to AL)
14900                                   <1>             ; AH = Negative AND mask (result must be ZERO)
14901                                   <1>     ; ESI = FindFile_Name address
14902                                   <1>
14903 00010BBE E8A77DFFFF              <1>     call   find_first_file
14904 00010BC3 72CA                    <1>     jc     short sysfff_err ; eax = 2 (File not found !)
14905                                   <1>
14906                                   <1>     ; ESI = Directory Entry (FindFile_DirEntry) Location
14907                                   <1>     ; EDI = Directory Buffer Directory Entry Location
14908                                   <1>     ; EAX = File Size
14909                                   <1>     ;  BL = Attributes of The File/Directory
14910                                   <1>     ;  BH = Long Name Yes/No Status (>0 is YES)
14911                                   <1>     ;  DX > 0 : Ambiguous filename chars are used
14912                                   <1>
14913                                   <1> sysfff_3:
14914                                   <1>     ; 16/10/2016
14915 00010BC5 668B0D[39830100]        <1>     mov    cx, [FFF_Attrib]
14916                                   <1>     ; Attribs in CL, return data type in CH
14917                                   <1>
14918                                   <1>     ;or    cl, cl
14919                                   <1>     ;jz    short sysfff_4 ; 0 = No filter
14920 00010BCC 80F1FF                  <1>     xor    cl, 0FFh
14921                                   <1>     ; cl = negative attributes ; 25/08/2024
14922 00010BCF 20D9                    <1>     and    cl, bl
14923 00010BD1 7409                    <1>     jz     short sysfff_4
14924                                   <1>
14925                                   <1>     ;mov   eax, 2 ; 'file not found !' error
14926                                   <1>     ;jmp   short sysfff_err_1
14927                                   <1>
14928                                   <1>     ; 16/10/2016
14929 00010BD3 E83F7EFFFF              <1>     call   find_next_file
14930 00010BD8 72B5                    <1>     jc     short sysfff_err ; eax = 12 (no more files !)
14931 00010BDA EBE9                    <1>     jmp    short sysfff_3
14932                                   <1>
14933                                   <1> sysfff_4:
14934 00010BDC 20ED                    <1>     and    ch, ch ; [FFF_RType]
14935                                   <1>     ;jz    short sysfff_5
14936                                   <1>     ; 25/08/2024
14937 00010BDE 755A                    <1>     jnz    short sysfff_7
14938                                   <1>
14939                                   <1>     ;mov   ecx, 128 ; ; transfer length
14940                                   <1>     ; 30/07/2022
14941                                   <1>     ;sub   ecx, ecx
14942                                   <1>     ;mov   cl, 128
14943                                   <1>     ;mov   [FFF_Valid], cl
14944                                   <1>     ; 25/08/2024
14945                                   <1>     ;mov   byte [FFF_Valid], 128 ; (*)
14946                                   <1>     ;jmp   short sysfff_7 ; sysfnf_11
14947                                   <1>
14948                                   <1> sysfff_5:
14949                                   <1>     ;;mov  esi, FindFile_DirEntry
14950                                   <1>     ;mov   ecx, 24  ; transfer length
14951                                   <1>     ; 30/07/2022
14952                                   <1>     ;sub   ecx, ecx
14953                                   <1>     ;mov   cl, 24
14954                                   <1>     ;mov   [FFF_Valid], cl
14955 00010BE0 C605[38830100]18        <1>     mov    byte [FFF_Valid], 24 ; (*)
14956                                   <1> sysfnf_12:
14957 00010BE7 BF[04880100]            <1>     mov    edi, DTA ; FFF data transfer address
14958                                   <1>     ;mov   al, [esi+DirEntry_Attr] ; 11
14959 00010BEC 88D8                    <1>     mov    al, bl ; File/Dir Attributes
```

```
14960 00010BEE 887F17              <1>     mov     [edi+23], bh ; Longname length (0= none)
14961 00010BF1 AA                  <1>     stosb
14962 00010BF2 88D0                <1>     mov     al, dl ; DL is for '?'
14963 00010BF4 00F0                <1>     add     al, dh ; DH is for '*'
14964                              <1>     ; AL > 0 if ambiguous file name wildcards are used
14965 00010BF6 AA                  <1>     stosb
14966 00010BF7 8B4616              <1>     mov     eax, [esi+DirEntry_WrtTime] ; 22
14967 00010BFA AB                  <1>         stosd       ; DirEntry_WrtTime & DirEntry_WrtDate
14968 00010BFB 8B461C              <1>         mov eax, [esi+DirEntry_FileSize] ; 28
14969 00010BFE AB                  <1>         stosd
14970 00010BFF 668B4614            <1>     mov     ax, [esi+DirEntry_FstClusHI] ; 20
14971                              <1>     ;shl    ax, 16
14972                              <1>     ; 23/07/2022 (BugFix)
14973 00010C03 C1E010              <1>     shl     eax, 16
14974 00010C06 668B461A            <1>     mov     ax, [esi+DirEntry_FstClusLO] ; 26
14975 00010C0A A3[348E0100]        <1>     mov     [u.r0], eax ; First Cluster
14976                              <1>
14977                              <1>         ;mov       esi, FindFile_DirEntry
14978 00010C0F E8A2110000          <1>     call    get_file_name
14979                              <1>
14980                              <1>     ; 25/08/2024
14981                              <1>     ; ecx <= 7 (from 'get_file_name')
14982                              <1>     ;mov    cl, [FFF_Valid] ; (*)
14983 00010C14 BE[04880100]        <1>             mov     esi, DTA ; FFF data transfer address
14984                              <1>
14985                              <1> sysfff_6:
14986                              <1>     ; 25/08/2024
14987                              <1>     ;sub    ecx, ecx
14988 00010C19 8A0D[38830100]      <1>     mov     cl, [FFF_Valid] ; (*) ecx <= 128
14989                              <1>
14990 00010C1F 8B3D[34830100]      <1>     mov     edi, [FFF_UBuffer] ; user's buffer address (edx)
14991 00010C25 E8C0FEFFFF          <1>     call    transfer_to_user_buffer
14992                              <1>
14993 00010C2A 890D[348E0100]      <1>     mov     [u.r0], ecx ; actual transfer count
14994 00010C30 E844110000          <1>         call       reset_working_path
14995 00010C35 E944BEFFFF          <1>     jmp     sysret
14996                              <1>
14997                              <1> sysfff_7:
14998                              <1>     ; 25/08/2024
14999 00010C3A C605[38830100]80    <1>     mov     byte [FFF_Valid], 128 ; (*)
15000                              <1> sysfnf_11:
15001                              <1>     ; ecx = 128
15002                              <1>     ; 25/08/2024 (bugfix)
15003                              <1>     ; 08/08/2022
15004 00010C41 BE[E27F0100]        <1>     mov     esi, FindFile_Drv
15005                              <1>     ; 25/08/2024
15006 00010C46 29C9                <1>     sub     ecx, ecx ; 0
15007 00010C48 EBCF                <1>     jmp     short sysfff_6
15008                              <1>
15009                              <1> sysfnf: ; <Find Next File>
15010                              <1>     ; 19/12/2025 (TRDOS 386 v2.0.10)
15011                              <1>     ; 25/08/2024 (TRDOS 386 Kernel v2.0.9)
15012                              <1>     ; 29/08/2023 (TRDOS 386 Kernel v2.0.6)
15013                              <1>     ; 08/08/2022
15014                              <1>     ; 30/07/2022 (TRDOS 386 Kernel v2.0.5)
15015                              <1>     ; 16/10/2016 TRDOS 386 (TRDOS v2.0) feature only !
15016                              <1>     ;           -derived from TRDOS v1.0, INT_21H.ASM-
15017                              <1>     ;            ("loc_INT21h_find_next_file")
15018                              <1>     ; TRDOS 8086 (v1.0)
15019                              <1>       ;  07/08/2011
15020                              <1>       ;  Find First File
15021                              <1>     ;       INPUT:
15022                              <1>       ;       none
15023                              <1>     ;       MSDOS OUTPUT:
15024                              <1>     ;           DTA: (Default address: PSP offset 80h)
15025                              <1>     ;           Offset  Descrription
15026                              <1>     ;           0       Reserved for use find next file
15027                              <1>     ;           21      Attribute of file found
15028                              <1>     ;           22      Time stamp of file
15029                              <1>     ;           24      Date stamp of file
15030                              <1>     ;           26      File size in bytes
15031                              <1>     ;           30      Filename and extension (zero terminated)
15032                              <1>     ;       If cf = 1:
15033                              <1>     ;           Error Codes: (in AX)
15034                              <1>     ;              18 - No more files
15035                              <1>       ;
15036                              <1>     ; TRDOS 386 (v2.0)
15037                              <1>     ; 16/10/2016
15038                              <1>     ;
15039                              <1>       ; INPUT ->
15040                              <1>           ;         none
15041                              <1>     ; OUTPUT ->
15042                              <1>     ;         EAX = 0 if CH input of 'Find First File' > 0
15043                              <1>     ;         EAX = First cluster number of file
15044                              <1>     ;               if CH input of 'Find First File' = 0
15045                              <1>     ;         EDX = File parameters table/structure address
15046                              <1>     ;
15047                              <1>     ;          cf = 1 -> Error code in AL
15048                              <1>     ;
15049                              <1>     ; Modified Registers: EAX (at the return of system call)
15050                              <1>
15051                              <1>     ; Note: If byte [FFF_Valid] = 0
15052                              <1>     ;       'sysfnf' will return with 'no more files' error.
15053                              <1>     ;       If byte [FFF_Valid] = 24
15054                              <1>     ;       'sysfnf' will return with 24 bytes basic parameters
15055                              <1>     ;       at the address which is in EDX.
15056                              <1>     ;       If byte [FFF_Valid] = 128
15057                              <1>     ;       'sysfnf' will return with 128 bytes Find File
15058                              <1>     ;       Structure/Table at the address which is in EDX.
15059                              <1>
15060 00010C4A 803D[38830100]00    <1>     cmp     byte [FFF_Valid], 0
15061 00010C51 7713                <1>     ja      short stsfnf_0
15062                              <1>     ; 'no more files !' error
15063                              <1>     ;mov    eax, ERR_NO_MORE_FILES ; 12
15064                              <1>     ; 30/07/2022
15065 00010C53 29C0                <1>     sub     eax, eax
15066 00010C55 B00C                <1>     mov     al, ERR_NO_MORE_FILES ; 12
15067 00010C57 A3[348E0100]        <1>     mov     [u.r0], eax
15068 00010C5C A3[A08E0100]        <1>     mov     [u.error], eax
15069 00010C61 E9F8BDFFFF          <1>     jmp     error
15070                              <1>
15071                              <1> stsfnf_0:
15072                              <1>     ;cmp    byte [FFF_Valid], 128
15073                              <1>     ;je     short stsfnf_1
15074                              <1>     ;cmp    byte [FFF_Valid], 24
15075                              <1>     ;je     short stsfnf_1
15076                              <1>     ;mov    [FFF_Valid], 24 ; Default
15077                              <1> stsfnf_1:
15078 00010C66 0FB61D[42770100]    <1>     movzx   ebx, byte [Current_Drv]
15079 00010C6D 66891D[3E830100]    <1>     mov     [SWP_DRV], bx
15080 00010C74 8A15[E27F0100]      <1>     mov     dl, [FindFile_Drv]
15081 00010C7A 38DA                <1>     cmp     dl, bl
15082 00010C7C 7532                <1>     jne     short stsfnf_2
15083 00010C7E 86DF                <1>     xchg    bh, bl
```

```
15084 00010C80 BE00010900          <1>     mov     esi, Logical_DOSDisks
15085 00010C85 01DE                <1>     add     esi, ebx
15086 00010C87 EB34                <1>     jmp     short sysfnf_3
15087                              <1>
15088                              <1> sysfnf_8:
15089 00010C89 E8A7B4FFFF          <1>     call    load_FAT_sub_directory
15090 00010C8E 7269                <1>     jc      short sysfnf_err_1 ; read error (no FNF stop)
15091                              <1>
15092                              <1> sysfnf_9:
15093 00010C90 E8827DFFFF          <1>     call    find_next_file
15094 00010C95 7257                <1>     jc      short sysfnf_5
15095                              <1>     ; 25/08/2024
15096                              <1>     ; esi = Directory Entry (FindFile_DirEntry) Location
15097                              <1>     ;; 08/08/2022
15098                              <1>     ;; esi = FindFile_Drv ; wrong ! ; 25/08/2024
15099                              <1>
15100 00010C97 A0[39830100]        <1>     mov     al, [FFF_Attrib]
15101                              <1>     ;or     al, al
15102                              <1>     ;jz     short sysfnf_10 ; 0 = No filter
15103 00010C9C 34FF                <1>     xor     al, 0FFh
15104 00010C9E 20D8                <1>     and     al, bl
15105 00010CA0 75EE                <1>     jnz     short sysfnf_9 ; search for next file until
15106                              <1>                            ; an error return from
15107                              <1>                            ; find_next_file procedure
15108                              <1> sysfnf_10:
15109                              <1>         ;movzx     ecx, byte [FFF_Valid]
15110                              <1>     ;cmp    cl, 128 ; complete FindFile structure/table
15111                              <1>     ;je     sysfnf_11
15112                              <1>     ;;cmp   cl, 24  ; basic parameters
15113                              <1>     ;;je    sysfnf_12
15114                              <1>     ;jmp    sysfnf_12
15115                              <1>     ; 30/07/2022
15116                              <1>     ;movzx  ecx, byte [FFF_Valid]
15117                              <1>     ; 25/08/2024
15118                              <1>     ;cmp    cl, 128
15119 00010CA2 803D[38830100]80    <1>     cmp     byte [FFF_Valid], 128
15120                              <1>     ;jne    short sysfnf_12
15121                              <1>     ;jmp    short sysfnf_11 (*)
15122                              <1>     ;; 08/08/2022
15123                              <1>     ;;je    short sysfnf_6 ; esi = FindFile_Drv
15124                              <1>     ; 29/08/2023 (BugFix)
15125                              <1>     ;je     short sysfff_6 ; esi = FindFile_Drv
15126                              <1>     ; 25/08/2024 (BugFix of BugFix) (*)
15127 00010CA9 7496                <1>     je      short sysfnf_11 ; esi <> FindFile_Drv
15128                              <1>
15129 00010CAB E937FFFFFF          <1>     jmp     sysfnf_12
15130                              <1>
15131                              <1> stsfnf_2:
15132 00010CB0 FE05[3F830100]      <1>     inc     byte [SWP_DRV_chg]
15133                              <1>
15134 00010CB6 E8256AFFFF          <1>     call    change_current_drive
15135 00010CBB 723C                <1>     jc      short sysfnf_err_1 ; read error !
15136                              <1>                                ; (do not stop, because
15137                              <1>                                ; we don't have a
15138                              <1>                                ; 'no more files'
15139                              <1>                                ; -file not found- error,
15140                              <1>                                ; next sysfnf system call
15141                              <1>                                ; may solve the problem,
15142                              <1>                                ; after re-placing the disk)
15143                              <1> sysfnf_3:
15144 00010CBD A1[58800100]        <1>     mov     eax, [FindFile_DirCluster]
15145 00010CC2 21C0                <1>     and     eax, eax
15146 00010CC4 7547                <1>     jnz     short sysfnf_6
15147                              <1>
15148 00010CC6 803D[41770100]02    <1>     cmp     byte [Current_FATType], 2
15149 00010CCD 7723                <1>     ja      short sysfnf_err_0 ; invalid, we neeed to stop !?
15150                              <1>
15151                              <1> ; 19/12/2025
15152                              <1> %if 0
15153                              <1>     cmp     byte [Current_FATType], 1
15154                              <1>     jb      short sysfnf_err_0 ; invalid, we neeed to stop !?
15155                              <1> %endif
15156                              <1>
15157 00010CCF 3805[697E0100]      <1>     cmp     byte [DirBuff_ValidData], al ; 0
15158 00010CD5 7608                <1>     jna     short sysfnf_4
15159                              <1>
15160 00010CD7 3B05[6E7E0100]      <1>     cmp     eax, [DirBuff_Cluster] ; 0 ?
15161 00010CDD 74B1                <1>     je      short sysfnf_9
15162                              <1>
15163                              <1>     ;cmp    byte [Current_Dir_Level], 0
15164                              <1>         ;ja short sysfnf_4
15165                              <1>         ;jna       short sysfnf_9
15166                              <1>
15167                              <1> sysfnf_4:
15168 00010CDF FE05[3F830100]      <1>     inc     byte [SWP_DRV_chg]
15169 00010CE5 E8CDB3FFFF          <1>     call    load_FAT_root_directory
15170 00010CEA 73A4                <1>     jnc     short sysfnf_9
15171                              <1>     ; eax = error code (17, 'drv not ready or read error')
15172 00010CEC EB0B                <1>     jmp     short sysfnf_err_1 ; read error ! (no FNF stop)
15173                              <1>                                ; (if you want, try again,
15174                              <1>                                ;  after re-placing the disk)
15175                              <1> sysfnf_5:
15176 00010CEE 3C0C                <1>     cmp     al, 12 ; 'no more files' error
15177 00010CF0 7507                <1>     jne     short sysfnf_err_1 ; (no FNF stop -sysfnf will try
15178                              <1>                                ;  to read the directory again,
15179                              <1>                                ;  if the user calls sysfnf
15180                              <1>                                ;  just after this error return-)
15181                              <1>     ; (FNF stop -sysfnf will not try
15182                              <1>     ;  to read the directory again-)
15183                              <1>
15184                              <1> sysfnf_err_0:
15185 00010CF2 C605[38830100]00    <1>     mov     byte [FFF_Valid], 0 ; FNF stop sign
15186                              <1> sysfnf_err_1:
15187 00010CF9 A3[348E0100]        <1>     mov     [u.r0], eax
15188 00010CFE A3[A08E0100]        <1>     mov     [u.error], eax
15189 00010D03 E871100000          <1>     call    reset_working_path
15190 00010D08 E951BDFFFF          <1>     jmp     error
15191                              <1>
15192                              <1> sysfnf_6:
15193 00010D0D 803D[697E0100]00    <1>     cmp     byte [DirBuff_ValidData], 0
15194 00010D14 760D                <1>     jna     short sysfnf_7
15195                              <1>
15196 00010D16 3B05[6E7E0100]      <1>     cmp     eax, [DirBuff_Cluster]
15197                              <1>     ;je     short sysfnf_9
15198                              <1>     ; 08/08/2022
15199 00010D1C 7505                <1>     jne     short sysfnf_7
15200 00010D1E E96DFFFFFF          <1>     jmp     sysfnf_9
15201                              <1> sysfnf_7:
15202 00010D23 FE05[3F830100]      <1>     inc     byte [SWP_DRV_chg]
15203                              <1>
15204                              <1> ; 19/12/2025
15205                              <1> %if 0
15206                              <1>     cmp     byte [Current_FATType], 1
15207                              <1>     ;jnb    short sysfnf_8
```

```
15208                                   <1>     ; 08/08/2022
15209                                   <1>     jb      short sysfnf_13
15210                                   <1> %endif
15211 00010D29 E95BFFFFFF               <1>     jmp     sysfnf_8
15212                                   <1>
15213                                   <1> ; 19/12/2025
15214                                   <1> %if 0
15215                                   <1> sysfnf_13:
15216                                   <1>     ; Singlix (TRFS) File System
15217                                   <1>     ; (access via compatibility buffer)
15218                                   <1>     call    load_FS_sub_directory
15219                                   <1>     ;jnc    short sysfnf_9
15220                                   <1>     ;jmp    short sysfnf_err_1 ; read error (no FNF stop)
15221                                   <1>     ; 08/08/2022
15222                                   <1>     jc      short sysfnf_err_1
15223                                   <1>     jmp     sysfnf_9
15224                                   <1> %endif
15225                                   <1>
15226                                   <1> writei:
15227                                   <1>     ; 02/09/2024
15228                                   <1>     ; 27/08/2024 - TRDOS 386 v2.0.9
15229                                   <1>     ; 08/08/2022
15230                                   <1>     ; 30/07/2022
15231                                   <1>     ; 23/07/2022 - TRDOS 386 Kernel v2.0.5
15232                                   <1>     ; 26/10/2016
15233                                   <1>     ; 25/10/2016
15234                                   <1>     ; 23/10/2016
15235                                   <1>     ; 22/10/2016
15236                                   <1>     ; 19/10/2016 - TRDOS 386 (TRDOS v2.0)
15237                                   <1>     ; 19/05/2015 - 20/05/2015 (Retro UNIX 386 v1)
15238                                   <1>     ; 12/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
15239                                   <1>     ;
15240                                   <1>     ; Write data to file with first cluster number in EAX
15241                                   <1>     ;
15242                                   <1>     ; INPUTS ->
15243                                   <1>     ;    EAX - First cluster number of the file
15244                                   <1>     ;    EBX - File number (Open file index number)
15245                                   <1>     ;    u.count - byte count to be written
15246                                   <1>     ;    u.base - points to user buffer
15247                                   <1>     ;    u.fofp - points to dword with current file offset
15248                                   <1>     ;    i.size - file size
15249                                   <1>     ;    cdev - logical dos drive number of the file
15250                                   <1>     ; OUTPUTS ->
15251                                   <1>     ;    u.count - cleared
15252                                   <1>     ;    u.nread - accumulates total bytes passed back
15253                                   <1>     ;    i.size - new file size (if file byte offset overs file size)
15254                                   <1>     ;    u.fofp - points to u.off (with new offset value)
15255                                   <1>     ;
15256                                   <1>     ; (Retro UNIX Prototype : 11/11/2012 - 18/11/2012, UNIXCOPY.ASM)
15257                                   <1>     ; ((Modified registers: eax, edx, ebx, ecx, esi, edi, ebp))
15258                                   <1>
15259 00010D2E 31C9                    <1>     xor     ecx, ecx
15260 00010D30 890D[608E0100]          <1>     mov     [u.nread], ecx  ; 0
15261 00010D36 66890D[988E0100]        <1>     mov     [u.pcount], cx ; 19/05/2015
15262 00010D3D 390D[5C8E0100]          <1>     cmp     [u.count], ecx
15263 00010D43 7701                    <1>     ja      short writei_1 ; 08/08/2022
15264 00010D45 C3                      <1>     retn
15265                                   <1>     ; 23/07/2022
15266                                   <1>     ;jna    short dskw_8 ; retn
15267                                   <1> writei_1:
15268                                   <1>     ; 02/09/2024
15269 00010D46 C705[DC820100]FFFF-      <1>     mov     dword [writei.fclust], 0FFFFFFFFh ; -1 ; reset
15269 00010D4E FFFF                     <1>
15270                                   <1> dskw:
15271 00010D50 881D[EC820100]          <1>     mov     [writei.ofn], bl ; Open file number
15272 00010D56 880D[2F830100]          <1>     mov     [setfmod], cl ; 0 ; reset 'update lm date&time' sign
15273                                   <1> dskw_0:
15274                                   <1>     ; 26/10/2016
15275                                   <1>     ; 22/10/2016, 23/10/2016, 25/10/2016
15276                                   <1>     ; 19/10/2016 - TRDOS 386 (TRDOS v2.0)
15277                                   <1>     ; 31/05/2015 - 25/07/2015 (Retro UNIX 386 v1)
15278                                   <1>     ; 26/04/2013 - 20/09/2013 (Retro UNIX 8086 v1)
15279                                   <1>     ;
15280                                   <1>     ; 01/08/2013 (mkdir_w check)
15281 00010D5C E8D0000000              <1>     call    mget_w
15282                                   <1>     ; eax = sector/block number
15283                                   <1>
15284 00010D61 8B1D[488E0100]          <1>     mov     ebx, [u.fofp]
15285 00010D67 8B13                    <1>     mov     edx, [ebx]
15286 00010D69 81E2FF010000            <1>     and     edx, 1FFh  ; / test the lower 9 bits of the file offset
15287 00010D6F 750C                    <1>     jnz     short dskw_1 ; / if its non-zero, branch
15288                                   <1>                          ; if zero, file offset = 0,
15289                                   <1>                          ; / 512, 1024,...(i.e., start of new block)
15290 00010D71 813D[5C8E0100]0002-      <1>     cmp     dword [u.count], 512
15290 00010D79 0000                     <1>
15291                                   <1>                            ; / if zero, is there enough data to fill
15292                                   <1>                            ; / an entire block? (i.e., no. of
15293 00010D7B 7331                    <1>     jnb     short dskw_2 ; / bytes to be written greater than 512.?
15294                                   <1>                          ; / Yes, branch. Don't have to read block
15295                                   <1> dskw_1: ; in as no past info. is to be saved
15296                                   <1>     ; (the entire block will be overwritten).
15297                                   <1>     ; 23/10/2016
15298                                   <1>
15299 00010D7D BB[64920100]            <1>     mov     ebx, writei_buffer
15300                                   <1>     ; esi = logical dos drive description table address
15301                                   <1>     ; eax = sector number
15302                                   <1>     ; ebx = buffer address (in kernel's memory space)
15303                                   <1>     ; ecx = sector count
15304                                   <1>     ; 30/07/2022
15305                                   <1>     ;mov    ecx, 1
15306 00010D82 31C9                    <1>     xor     ecx, ecx
15307 00010D84 FEC1                    <1>     inc     cl
15308                                   <1>     ; ecx = 1
15309 00010D86 E8370C0000              <1>     call    disk_read
15310                                   <1>     ;call   dskrd   ; / no, must retain old info..
15311                                   <1>                     ; / Hence, read block 'r1' into an I/O buffer
15312 00010D8B 7321                    <1>     jnc     short dskw_2
15313                                   <1>
15314                                   <1>     ; disk read error
15315                                   <1>     ; 30/07/2022
15316                                   <1>     ;mov    eax, 17 ; drive not ready or READ ERROR !
15317 00010D8D 29C0                    <1>     sub     eax, eax
15318 00010D8F B011                    <1>     mov     al, 17
15319                                   <1> dskw_err: ; jump from disk write error
15320 00010D91 A3[348E0100]            <1>     mov     [u.r0], eax
15321 00010D96 A3[A08E0100]            <1>     mov     [u.error], eax
15322                                   <1>
15323 00010D9B 803D[2F830100]00        <1>     cmp     byte [setfmod], 0
15324                                   <1>     ;jna    error
15325                                   <1>     ; 23/07/2022
15326 00010DA2 7605                    <1>     jna     short writei_err
15327                                   <1>
15328 00010DA4 E887020000              <1>     call    update_file_lmdt ; update last modif. date&time of the file
15329                                   <1>     ;mov    byte [setfmod], 0
```

```
15330                                   <1> writei_err:
15331 00010DA9 E9B0BCFFFF               <1>     jmp     error
15332                                   <1>
15333                                   <1> dskw_2: ; 3:
15334                                   <1>     ; 23/10/2016
15335 00010DAE C605[C8820100]01         <1>     mov     byte [writei.valid], 1 ; writei buffer contains valid data
15336 00010DB5 56                       <1>     push    esi ; logical dos drive description table address
15337                                   <1>     ; EAX (r1) = block/sector number
15338                                   <1>     ;call  wslot
15339                                   <1>             ; jsr r0,wslot / set write and inhibit bits in I/O queue,
15340                                   <1>                     ; / proc. status=0, r5 points to 1st word of data
15341 00010DB6 803D[9A8E0100]00         <1>     cmp     byte [u.kcall], 0
15342 00010DBD 770F                     <1>     ja      short dskw_4 ; zf=0 -> the caller is 'mkdir'
15343                                   <1>     ;
15344 00010DBF 66833D[988E0100]00       <1>     cmp     word [u.pcount], 0
15345 00010DC7 7705                     <1>     ja      short dskw_4
15346                                   <1> dskw_3:
15347                                   <1>     ; [u.base] = virtual address to transfer (as source address)
15348 00010DC9 E844FAFFFF               <1>     call    trans_addr_r ; translate virtual address to physical (r)
15349                                   <1> dskw_4:
15350 00010DCE BB[64920100]             <1>     mov     ebx, writei_buffer
15351                                   <1>     ; EBX (r5) = system (I/O) buffer address
15352 00010DD3 E8A6FAFFFF               <1>     call    sioreg
15353                                   <1>     ; ESI = file (user data) offset
15354                                   <1>     ; EDI = sector (I/O) buffer offset
15355                                   <1>     ; ECX = byte count
15356                                   <1>     ;
15357 00010DD8 F3A4                     <1>     rep     movsb
15358                                   <1>     ; 25/07/2015
15359                                   <1>     ; eax = remain bytes in buffer
15360                                   <1>        ;       (check if remain bytes in the buffer > [u.pcount])
15361 00010DDA 09C0                     <1>     or      eax, eax
15362 00010DDC 75EB                     <1>     jnz     short dskw_3 ; (page end before system buffer end!)
15363                                   <1>
15364                                   <1>     ; 23/10/2016
15365 00010DDE B101                     <1>     mov     cl, 1
15366 00010DE0 5E                       <1>     pop     esi
15367 00010DE1 A1[CC820100]             <1>     mov     eax, [writei.sector]
15368                                   <1>     ; esi = logical dos drive description table address
15369                                   <1>     ; eax = sector number
15370                                   <1>     ; ebx = writei buffer address
15371                                   <1>     ; ecx = sector count
15372 00010DE6 E8C80B0000               <1>     call    disk_write ; / yes, write the block
15373 00010DEB 7313                     <1>     jnc     short dskw_5
15374                                   <1>
15375                                   <1>     ;mov    eax, 18 ; drive not ready or WRITE ERROR !
15376                                   <1>     ; 30/08/2022
15377 00010DED 29C0                     <1>     sub     eax, eax
15378 00010DEF B012                     <1>     mov     al, 18
15379 00010DF1 EB9E                     <1>     jmp     short dskw_err
15380                                   <1>
15381                                   <1> dskw_7:
15382                                   <1>     ; update last modif. date&time of the file
15383                                   <1>     ; (also updates file size as OF_SIZE)
15384 00010DF3 E838020000               <1>     call    update_file_lmdt
15385                                   <1>     ;mov    byte [setfmod], 0
15386                                   <1>
15387                                   <1>     ; 03/08/2013
15388 00010DF8 C605[9A8E0100]00         <1>     mov     byte [u.kcall], 0
15389                                   <1>     ; 23/10/2016
15390                                   <1>     ;mov    eax, [writei.fclust]
15391                                   <1> dskw_8:             ; 23/07/2022
15392 00010DFF C3                       <1>     retn
15393                                   <1>
15394                                   <1> dskw_5:
15395                                   <1>     ; 26/10/2016
15396 00010E00 0FB61D[EC820100]         <1>     movzx   ebx, byte [writei.ofn] ; open file number
15397 00010E07 C0E302                   <1>     shl     bl, 2 ; *4
15398 00010E0A 8B83[44840100]           <1>     mov     eax, [ebx+OF_POINTER]
15399 00010E10 3B83[C4840100]           <1>     cmp     eax, [ebx+OF_SIZE]
15400 00010E16 7606                     <1>     jna     short dskw_6
15401 00010E18 8983[C4840100]           <1>     mov     [ebx+OF_SIZE], eax
15402                                   <1> dskw_6:
15403                                   <1>     ;shr    bl, 2
15404 00010E1E 833D[5C8E0100]00         <1>         cmp     dword [u.count], 0 ; / any more data to write?
15405 00010E25 76CC                     <1>     jna     short dskw_7
15406 00010E27 A1[DC820100]             <1>     mov     eax, [writei.fclust]
15407 00010E2C E92BFFFFFF               <1>     jmp     dskw_0 ; / yes, branch
15408                                   <1>
15409                                   <1> mget_w:
15410                                   <1>     ; 19/12/2025 - TRDOS 386 v2.0.10
15411                                   <1>     ; 03/09/2024
15412                                   <1>     ; 02/09/2024
15413                                   <1>     ; 25/08/2024 - TRDOS 386 v2.0.9
15414                                   <1>     ; 08/08/2022
15415                                   <1>     ; 25/07/2022
15416                                   <1>     ; 23/07/2022 - TRDOS 386 Kernel v2.0.5
15417                                   <1>     ; 02/11/2016
15418                                   <1>     ; 01/11/2016
15419                                   <1>     ; 23/10/2016, 31/10/2016
15420                                   <1>     ; 22/10/2016 - TRDOS 386 (TRDOS v2.0)
15421                                   <1>     ; 03/06/2015 (Retro UNIX 386 v1, 'mget', u.5s)
15422                                   <1>     ; 22/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
15423                                   <1>     ;
15424                                   <1>     ; Get existing or (allocate) a new disk block for file
15425                                   <1>     ;
15426                                   <1>     ; INPUTS ->
15427                                   <1>     ;    [u.fofp] = file offset pointer
15428                                   <1>     ;    [i.size] = file size
15429                                   <1>     ;    [u.count] = byte count
15430                                   <1>     ;    EAX = First cluster
15431                                   <1>     ;    [cdev] = Logical dos drive number
15432                                   <1>     ;    [writei.ofn] = File Number
15433                                   <1>     ;                   (Open file index, 0 based)
15434                                   <1>     ;    ([u.off] = file offset)
15435                                   <1>     ; OUTPUTS ->
15436                                   <1>     ;    EAX = logical sector number
15437                                   <1>     ;    ESI = Logical Dos Drive Description Table address
15438                                   <1>     ;
15439                                   <1>     ; Modified registers: EDX, EBX, ECX, ESI, EDI, EBP
15440                                   <1>
15441 00010E31 8B35[488E0100]           <1>         mov     esi, [u.fofp]
15442 00010E37 8B2E                     <1>     mov     ebp, [esi] ; u.off (or EBX*4+OF_POINTER)
15443                                   <1>
15444 00010E39 29C9                     <1>     sub     ecx, ecx
15445 00010E3B 8A2D[198E0100]           <1>     mov     ch, [cdev]
15446                                   <1>
15447 00010E41 BE00010900               <1>     mov     esi, Logical_DOSDisks
15448 00010E46 01CE                     <1>     add     esi, ecx
15449                                   <1>
15450                                   <1>     ; 31/10/2016
15451 00010E48 89C3                     <1>     mov     ebx, eax ; First Cluster or FDT address
15452                                   <1>
15453                                   <1> ; 19/12/2025
```

```
15454                                  <1> %if 0
15455                                  <1>     cmp     byte [esi+LD_FATType], 0
15456                                  <1>     ;jna    mget_w_14 ; Singlix FS
15457                                  <1>     ; 23/07/2022
15458                                  <1>     ja      short mget_w_20
15459                                  <1>     jmp     mget_w_14 ; Singlix FS
15460                                  <1> %endif
15461                                  <1>
15462                                  <1> mget_w_20:
15463                                  <1>     ;movzx  eax, word [esi+LD_BPB+BytesPerSec]
15464                                  <1>     ; 19/12/2025
15465 00010E4A B800020000               <1>     mov     eax, 512
15466 00010E4F 0FB65613                 <1>     movzx   edx, byte [esi+LD_BPB+SecPerClust]
15467 00010E53 8815[CA820100]           <1>     mov     [writei.spc], dl  ; sectors per cluster
15468 00010E59 F7E2                     <1>     mul     edx
15469                                  <1>     ; edx = 0
15470                                  <1>     ; eax = bytes per cluster (<= 65536)
15471                                  <1>
15472                                  <1>     ; 02/11/2016
15473 00010E5B 89C1                     <1>     mov     ecx, eax
15474 00010E5D 48                       <1>     dec     eax
15475 00010E5E 66A3[D0820100]           <1>     mov     [writei.bpc], ax
15476                                  <1>
15477 00010E64 89E8                     <1>     mov     eax, ebp
15478 00010E66 0305[5C8E0100]           <1>     add     eax, [u.count] ; next file position
15479 00010E6C 3B05[408F0100]           <1>     cmp     eax, [i.size] ; <= file size ?
15480                                  <1>     ;jna    mget_w_4 ; no
15481                                  <1>     ; 23/07/2022
15482 00010E72 7705                     <1>     ja      short mget_w_21
15483                                  <1>     ; 02/09/2024
15484                                  <1>     ; ebx  = first cluster (input)
15485 00010E74 E925010000               <1>     jmp     mget_w_4
15486                                  <1> mget_w_21:
15487 00010E79 F7F1                     <1>     div     ecx
15488 00010E7B A3[D8820100]             <1>     mov     [writei.c_index], eax ; cluster index
15489                                  <1>     ; edx = byte offset in cluster (<= 65535)
15490                                  <1>     ;mov    [writei.offset], dx
15491                                  <1>     ;shr    dx, 9 ; / 512
15492                                  <1>     ;mov    [writei.s_index], dl ; sector index in cluster (0 to spc -1)
15493                                  <1>
15494 00010E80 29D2                     <1>     sub     edx, edx ; 01/11/2016
15495 00010E82 8915[CC820100]           <1>     mov     [writei.sector], edx ; 0
15496 00010E88 668915[D2820100]         <1>     mov     [writei.offset], dx  ; byte offset in cluster
15497 00010E8F 8815[CB820100]           <1>     mov     [writei.s_index], dl ; sector index in cluster (0 to spc -1)
15498                                  <1>
15499 00010E95 89D8                     <1>     mov     eax, ebx ; First Cluster
15500                                  <1>
15501                                  <1>     ; is this the 1st mget_w or a next mget_w call ? (by 'writei')
15502 00010E97 3815[C8820100]           <1>     cmp     byte [writei.valid], dl ; 0
15503 00010E9D 7624                     <1>     jna     short mget_w_0
15504                                  <1>
15505 00010E9F 8815[C8820100]           <1>     mov     byte [writei.valid], dl ; 0 ; reset ('writei' will set it)
15506                                  <1>
15507 00010EA5 3B05[DC820100]           <1>     cmp     eax, [writei.fclust]
15508 00010EAB 7516                     <1>     jne     short mget_w_0
15509                                  <1>
15510 00010EAD 8A0D[198E0100]           <1>     mov     cl, [cdev]
15511 00010EB3 3A0D[C9820100]           <1>     cmp     cl, [writei.drv]
15512 00010EB9 7508                     <1>     jne     short mget_w_0
15513                                  <1>     ; [writei.l_clust] & [writei.l_index] are valid,
15514                                  <1>     ;  we don't need to get last cluster & last cluster index
15515 00010EBB 8B0D[E8820100]           <1>     mov     ecx, [writei.l_index]
15516 00010EC1 EB7F                     <1>     jmp     short mget_w_2
15517                                  <1> mget_w_0:
15518 00010EC3 A3[DC820100]             <1>     mov     [writei.fclust], eax ; first cluster
15519                                  <1>     ; edx = 0
15520 00010EC8 A3[D4820100]             <1>     mov     [writei.cluster], eax ; first cluster ; 01/11/2016
15521 00010ECD 8915[E0820100]           <1>     mov     [writei.fs_index], edx ; 0 ; current cluster index
15522                                  <1>
15523                                  <1>     ;;;
15524                                  <1>     ; 25/08/2024 - TRDOS 386 v2.0.9
15525 00010ED3 09C0                     <1>     or      eax, eax ; is first cluster number = 0 ?
15526 00010ED5 751A                     <1>     jnz     short mget_w_27 ; no
15527                                  <1>
15528                                  <1>     ; eax = 0 (*)
15529                                  <1>     ; ((a directory entry with zero file size)) (*)
15530                                  <1>     ; (*) ((('syswrite' may be used for writing to an empty file.
15531                                  <1>     ; Before this modification, 'syscreat' and 'syswrite' system calls
15532                                  <1>     ; did wrong things together... 'writetest.s' file, 24/08/2024.)))
15533                                  <1>
15534 00010ED7 E820B9FFFF              <1>     call    add_new_cluster ; (*) 2024 (v2.0.9) modification
15535 00010EDC 721A                     <1>     jc      short mget_w_errj ; eax = error code
15536                                  <1>     ; eax = (new) first cluster
15537                                  <1>     ; skip 'get_last_cluster' and set ecx to -1
15538                                  <1>     ;
15539                                  <1>     ; 02/09/2024
15540 00010EDE A3[DC820100]             <1>     mov     [writei.fclust], eax ; first cluster
15541 00010EE3 A3[D4820100]             <1>     mov     [writei.cluster], eax
15542                                  <1>     ; 03/09/2024
15543 00010EE8 A3[30830100]             <1>     mov     [setfclust], eax
15544                                  <1>     ;
15545 00010EED 31C9                     <1>     xor     ecx, ecx
15546 00010EEF EB12                     <1>     jmp     short mget_w_28
15547                                  <1>     ;;;
15548                                  <1>
15549                                  <1> mget_w_27:
15550                                  <1>     ; FAT file system (FAT12, FAT16, FAT32)
15551 00010EF1 E8E9B7FFFF              <1>     call    get_last_cluster
15552                                  <1>     ;jc     short mget_w_err ; eax = error code
15553                                  <1>     ; 08/08/2022
15554 00010EF6 7305                     <1>     jnc     short mget_w_25
15555                                  <1> mget_w_errj:       ; 25/08/2024
15556 00010EF8 E992000000               <1>     jmp     mget_w_err
15557                                  <1>
15558                                  <1> mget_w_25:
15559                                  <1>     ; 25/08/2024
15560 00010EFD 8B0D[10810100]           <1>     mov     ecx, [glc_index] ; last cluster index
15561                                  <1> mget_w_28:  ; 02/09/2024
15562 00010F03 890D[E8820100]           <1>     mov     [writei.l_index], ecx
15563 00010F09 A3[E4820100]             <1>     mov     [writei.lclust], eax ; last cluster
15564                                  <1>
15565 00010F0E A0[EC820100]             <1>     mov     al, [writei.ofn]
15566 00010F13 FEC0                     <1>     inc     al
15567 00010F15 A2[2F830100]             <1>     mov     [setfmod], al ; update lm date&time sign
15568                                  <1>
15569                                  <1> mget_w_1:
15570 00010F1A 3B0D[D8820100]           <1>     cmp     ecx, [writei.c_index] ; last cluster index
15571 00010F20 7320                     <1>     jnb     short mget_w_2 ; 01/11/2016
15572                                  <1>
15573 00010F22 A1[E4820100]             <1>     mov     eax, [writei.lclust]
15574                                  <1>     ; EAX = Last cluster
15575 00010F27 E8D0B8FFFF              <1>     call    add_new_cluster
15576 00010F2C 7261                     <1>     jc      short mget_w_err ; eax = error code
15577                                  <1>     ; edx = 0
```

```
15578 00010F2E A3[E4820100]       <1>     mov     [writei.lclust], eax ; (new) last cluster
15579 00010F33 8B0D[E8820100]     <1>     mov     ecx, [writei.l_index]
15580 00010F39 41                 <1>     inc     ecx ; add 1 to last cluster index
15581 00010F3A 890D[E8820100]     <1>     mov     [writei.l_index], ecx ; current last cluster index
15582                             <1>
15583 00010F40 EBD8               <1>     jmp     short mget_w_1
15584                             <1>
15585                             <1> mget_w_2:
15586 00010F42 89E9               <1>     mov     ecx, ebp
15587 00010F44 030D[5C8E0100]     <1>     add     ecx, [u.count]
15588 00010F4A 890D[408F0100]     <1>     mov     [i.size], ecx ; save new file size
15589                             <1>     ;sub    edx, edx ; 0
15590                             <1>
15591 00010F50 A0[198E0100]       <1>     mov     al, [cdev]
15592 00010F55 A2[C9820100]       <1>     mov     [writei.drv], al ; physical drive number
15593                             <1>     ; edx = 0
15594 00010F5A 89E8               <1>     mov     eax, ebp ; file offset
15595 00010F5C 0FB70D[D0820100]   <1>     movzx   ecx, word [writei.bpc] ; bytes per cluster - 1
15596 00010F63 41                 <1>     inc     ecx ; bytes per cluster
15597 00010F64 F7F1               <1>     div     ecx
15598                             <1>     ; edx = byte offset in cluster (<= 65535)
15599                             <1>     ; eax = cluster index
15600 00010F66 A3[D8820100]       <1>     mov     [writei.c_index], eax
15601 00010F6B 668915[D2820100]   <1>     mov     [writei.offset], dx
15602                             <1>     ;shr    dx, 9 ; / 512
15603                             <1>     ; 23/07/2022
15604 00010F72 C1EA09             <1>     shr     edx, 9
15605 00010F75 8815[CB820100]     <1>     mov     [writei.s_index], dl ; sector index in cluster (0 to spc -1)
15606                             <1>
15607                             <1> mget_w_3:
15608 00010F7B 3B05[E8820100]     <1>     cmp     eax, [writei.l_index] ; last cluster index
15609 00010F81 753E               <1>     jne     short mget_w_5
15610                             <1>
15611 00010F83 A3[E0820100]       <1>     mov     [writei.fs_index], eax ; cluster index (for next check)
15612                             <1>
15613 00010F88 A1[E4820100]       <1>     mov     eax, [writei.lclust] ; last cluster
15614 00010F8D EB74               <1>     jmp     short mget_w_10
15615                             <1>
15616                             <1> mget_w_err:
15617 00010F8F A3[A08E0100]       <1>     mov     [u.error], eax
15618 00010F94 A3[348E0100]       <1>     mov     [u.r0], eax
15619 00010F99 E9C0BAFFFF         <1>     jmp     error
15620                             <1>
15621                             <1> mget_w_4: ; 02/11/2016
15622                             <1>     ; eax = next file position
15623 00010F9E 2B05[5C8E0100]     <1>     sub     eax, [u.count] ; current file position
15624                             <1>     ; edx = 0
15625                             <1>     ; ecx = bytes per cluster
15626 00010FA4 F7F1               <1>     div     ecx
15627 00010FA6 A3[D8820100]       <1>     mov     [writei.c_index], eax ; cluster index
15628 00010FAB 668915[D2820100]   <1>     mov     [writei.offset], dx
15629                             <1>     ;shr    dx, 9 ; / 512
15630                             <1>     ; 23/07/2022
15631 00010FB2 C1EA09             <1>     shr     edx, 9
15632 00010FB5 8815[CB820100]     <1>     mov     [writei.s_index], dl ; sector index in cluster (0 to spc -1)
15633                             <1>
15634                             <1>     ; 02/09/2024
15635 00010FBB 891D[DC820100]     <1>     mov     [writei.fclust], ebx
15636                             <1>
15637                             <1> mget_w_5:
15638 00010FC1 21C0               <1>     and     eax, eax ; 0 = First Cluster's index number
15639 00010FC3 750C               <1>     jnz     short mget_w_6
15640                             <1>
15641 00010FC5 A3[E0820100]       <1>     mov     [writei.fs_index], eax ; cluster index (for next check)
15642                             <1>
15643 00010FCA A1[DC820100]       <1>     mov     eax, [writei.fclust] ; first cluster
15644 00010FCF EB32               <1>     jmp     short mget_w_10
15645                             <1>
15646                             <1> mget_w_6:
15647 00010FD1 3B05[E0820100]     <1>     cmp     eax, [writei.fs_index] ; current cluster index (>0)
15648 00010FD7 7507               <1>     jne     short mget_w_7
15649 00010FD9 A1[D4820100]       <1>     mov     eax, [writei.cluster] ; current cluster
15650 00010FDE EB34               <1>     jmp     short mget_w_11
15651                             <1>
15652                             <1> mget_w_7:
15653 00010FE0 89C1               <1>     mov     ecx, eax
15654 00010FE2 2B0D[E0820100]     <1>     sub     ecx, [writei.fs_index]
15655 00010FE8 730D               <1>     jnc     short mget_w_8
15656                             <1>     ; get cluster by index from the first cluster
15657 00010FEA A1[DC820100]       <1>     mov     eax, [writei.fclust]
15658 00010FEF 8B0D[D8820100]     <1>     mov     ecx, [writei.c_index]
15659 00010FF5 EB05               <1>     jmp     short mget_w_9
15660                             <1>
15661                             <1> mget_w_8:
15662 00010FF7 A1[D4820100]       <1>     mov     eax, [writei.cluster] ; beginning cluster
15663                             <1>     ; ecx = cluster sequence number after the beginning cluster
15664                             <1>     ; sub   edx, edx ; 0
15665                             <1>
15666                             <1> mget_w_9:
15667                             <1>     ; EAX = Beginning cluster
15668                             <1>     ; EDX = Sector index in disk/file section
15669                             <1>     ;       (Only for SINGLIX file system!)
15670                             <1>     ; ECX = Cluster sequence number after the beginning cluster
15671                             <1>     ; ESI = Logical DOS Drive Description Table address
15672 00010FFC E8A8B8FFFF         <1>     call    get_cluster_by_index
15673 00011001 728C               <1>     jc      short mget_w_err ; error code in EAX
15674                             <1>
15675                             <1>     ; EAX = Cluster number
15676                             <1> mget_w_10:
15677 00011003 A3[D4820100]       <1>     mov     [writei.cluster], eax ; FDT number for Singlix File System
15678                             <1>
15679                             <1> ; 19/12/2025
15680                             <1> %if 0
15681                             <1>     cmp     byte [esi+LD_FATType], 0
15682                             <1>     jna     short mget_w_13
15683                             <1> %endif
15684                             <1>     ; 01/11/2016
15685 00011008 8B15[D8820100]     <1>     mov     edx, [writei.c_index]
15686 0001100E 8915[E0820100]     <1>     mov     [writei.fs_index], edx
15687                             <1> mget_w_11:
15688                             <1>     ;sub    eax, 2
15689                             <1>     ; 23/07/2022
15690 00011014 48                 <1>     dec     eax
15691 00011015 48                 <1>     dec     eax
15692 00011016 0FB615[CA820100]   <1>     movzx   edx, byte [writei.spc]
15693 0001101D F7E2               <1>     mul     edx
15694                             <1>
15695 0001101F 034668             <1>     add     eax, [esi+LD_DATABegin]
15696 00011022 8A15[CB820100]     <1>     mov     dl, [writei.s_index]
15697 00011028 01D0               <1>     add     eax, edx
15698                             <1> mget_w_12:
15699 0001102A A3[CC820100]       <1>     mov     [writei.sector], eax
15700                             <1>     ;; buffer validation must be done in writei
15701                             <1>     ;;mov   byte [writei.valid], 1
```

```
15702 0001102F C3                   <1>     retn
15703                               <1>
15704                               <1> ; 19/12/2025
15705                               <1> %if 0
15706                               <1>
15707                               <1> mget_w_13:
15708                               <1>     ; EAX = FDT number (Current Section)
15709                               <1>     ; EDX = Sector index from the first section (0,1,2,3,4...)
15710                               <1>     sub     edx, [writei.fs_index]
15711                               <1>     ; EDX = Sector index from current section
15712                               <1>     mov     [writei.fs_index], edx
15713                               <1>     inc     eax ; the first data sector in FS disk section
15714                               <1>     add     eax, edx
15715                               <1>     jmp     short mget_w_12
15716                               <1>
15717                               <1> mget_w_14:
15718                               <1>     ;mov    cl, [esi+LD_FS_BytesPerSec+1]
15719                               <1>     ;shr    cl, 1 ;  ; 1 for 512 bytes, 4 for 2048 bytes
15720                               <1>     ;mov    [writei.spc], cl  ; sectors per cluster
15721                               <1>     ; 19/12/2025
15722                               <1>     mov     byte [writei.spc], 1
15723                               <1>     ; NOTE: writei bytes per sector value is always 512 !
15724                               <1>     mov     word [writei.bpc], 512
15725                               <1>
15726                               <1>     mov     ecx, ebp
15727                               <1>     add     ecx, [u.count] ; next file position
15728                               <1>     cmp     ecx, [i.size] ; <= file size ?
15729                               <1>     ;jna    mget_w_19 ; no
15730                               <1>     ja      short mget_w_22
15731                               <1>     ; 23/07/2022
15732                               <1>     jmp     mget_w_19
15733                               <1>
15734                               <1> mget_w_22:
15735                               <1>     sub     edx, edx ; 0
15736                               <1>     mov     [writei.sector], edx ; 0
15737                               <1>     mov     [writei.offset], dx  ; byte offset in cluster
15738                               <1>     mov     [writei.s_index], dl ; sector index in cluster (0 to spc -1)
15739                               <1>
15740                               <1>     shr     ecx, 9 ; 1 cluster = 512 bytes
15741                               <1>     mov     [writei.c_index], ecx ; section/cluster index
15742                               <1>
15743                               <1>     mov     eax, ebx ; FDT number (First FDT address)
15744                               <1>
15745                               <1>     ; is this the 1st mget_w or a next mget_w call ? (by 'writei')
15746                               <1>     cmp     byte [writei.valid], dl ; 0
15747                               <1>     jna     short mget_w_15
15748                               <1>
15749                               <1>     mov     byte [writei.valid], dl ; 0 ; reset ('writei' will set it)
15750                               <1>
15751                               <1>     cmp     eax, [writei.fclust]
15752                               <1>     jne     short mget_w_15
15753                               <1>
15754                               <1>     mov     cl, [cdev]
15755                               <1>     cmp     cl, [writei.drv]
15756                               <1>     jne     short mget_w_15
15757                               <1>     ; [writei.l_clust] & [writei.l_index] are valid,
15758                               <1>     ;  we don't need to get last cluster & last cluster index
15759                               <1>     mov     ecx, [writei.l_index]
15760                               <1>     jmp     short mget_w_17
15761                               <1> mget_w_15:
15762                               <1>     mov     [writei.fclust], eax ; first section (FDT number)
15763                               <1>     ; edx = 0
15764                               <1>     mov     [writei.cluster], edx ; 0 ; current section
15765                               <1>     mov     [writei.fs_index], edx ; 0 ; curret section index
15766                               <1>
15767                               <1>     ; eax = FDT number (section 0 header address)
15768                               <1>     call    get_last_section
15769                               <1>     ;jc     short mget_w_err ; eax = error code
15770                               <1>     ; 08/08/2022
15771                               <1>     jnc     short mget_w_26
15772                               <1>     jmp     mget_w_err
15773                               <1> mget_w_26:
15774                               <1>     mov     [writei.fs_index], edx ; sector index in last section
15775                               <1>
15776                               <1>     mov     [writei.lclust], eax ; last section address
15777                               <1>
15778                               <1>     mov     ecx, [glc_index] ; last section index
15779                               <1>     mov     [writei.l_index], ecx
15780                               <1>
15781                               <1>     mov     al, [writei.ofn]
15782                               <1>     inc     al
15783                               <1>     mov     [setfmod], al ; update lm date&time sign
15784                               <1>
15785                               <1> mget_w_16:
15786                               <1>     ; edx = (existing) last section (sector) index
15787                               <1>     mov     ecx, [writei.c_index] ; final section (sector) index
15788                               <1>     sub     ecx, edx
15789                               <1>     jna     short mget_w_19
15790                               <1>     ; ecx = sector count
15791                               <1> mget_w_17:
15792                               <1>     mov     eax, [writei.lclust]
15793                               <1>     ; ESI = Logical dos drv desc. table address
15794                               <1>         ; EAX = Last section
15795                               <1>         ; (ECX = 0 for directory)
15796                               <1>         ; ECX = sector count (except FDT)
15797                               <1>     call    add_new_fs_section
15798                               <1>     jnc     short mget_w_18
15799                               <1>
15800                               <1>     ; If error number = 27h (insufficient disk space)
15801                               <1>     ; it is needed to check free consequent sectors
15802                               <1>     ; (1 data sector at least and +1 section header sector)
15803                               <1>
15804                               <1>     cmp     eax, 27h
15805                               <1>     ;jne    mget_w_err ; eax = error code
15806                               <1>     ; 25/07/2022
15807                               <1>     je      short mget_w_24
15808                               <1> mget_w_23:
15809                               <1>     jmp     mget_w_err
15810                               <1> mget_w_24:
15811                               <1>     ; ecx = count of free consequent sectors
15812                               <1>     ; ecx must be > 1 (1 data + 1 header sector)
15813                               <1>     dec     ecx
15814                               <1>     ;jz     short mget_w_err
15815                               <1>     ;jmp    short mget_w_17
15816                               <1>     ; 23/07/2022
15817                               <1>     jnz     short mget_w_17
15818                               <1>     ;jmp    mget_w_err
15819                               <1>     ; 25/07/2022
15820                               <1>     jmp     short mget_w_23
15821                               <1>
15822                               <1> mget_w_18:
15823                               <1>     mov     [writei.lclust], eax ; (new) last section
15824                               <1>     ; ecx = sector count (except section header)
15825                               <1>     mov     edx, [writei.l_index]
```

```
15826                                   <1>     add     edx, ecx ; add sector count to index
15827                                   <1>     mov     [writei.l_index], edx
15828                                   <1>     jmp     short mget_w_16
15829                                   <1>
15830                                   <1> mget_w_19:
15831                                   <1>     mov     ecx, ebp
15832                                   <1>     add     ecx, [u.count]
15833                                   <1>     mov     [i.size], ecx ; save new file size
15834                                   <1>     ;sub    edx, edx ; 0
15835                                   <1>
15836                                   <1>     mov     al, [cdev]
15837                                   <1>     mov     [writei.drv], al ; physical drive number
15838                                   <1>     ; edx = 0
15839                                   <1>     mov     eax, ebp ; file offset
15840                                   <1>     mov     edx, eax
15841                                   <1>     ; 1 cluster = 512 bytes (for Singlix FS)
15842                                   <1>     shr     eax, 9  ; / 512
15843                                   <1>     and     edx, 1FFh
15844                                   <1>     ; edx = byte offset in cluster/sector (<= 511)
15845                                   <1>     ; eax = section (sector/cluster) index
15846                                   <1>     mov     [writei.c_index], eax
15847                                   <1>     mov     [writei.offset], dx
15848                                   <1>     ;mov    byte [writei.s_index], 0 ; sector index in cluster
15849                                   <1>     jmp     mget_w_3
15850                                   <1>
15851                                   <1> %endif
15852                                   <1>
15853                                   <1> update_file_lmdt: ; & update file size
15854                                   <1>     ; 19/12/2025 - TRDOS 386 v2.0.10
15855                                   <1>     ; 03/09/2024
15856                                   <1>     ; 27/08/2024 - TRDOS 386 v2.0.9
15857                                   <1>     ; 30/07/2022
15858                                   <1>     ; 23/07/2022 - TRDOS 386 v2.0.5
15859                                   <1>     ; 26/10/2016
15860                                   <1>     ; 24/10/2016
15861                                   <1>     ; 23/10/2016
15862                                   <1>     ; 22/10/2016 - TRDOS 386 (TRDOS v2.0)
15863                                   <1>     ;
15864                                   <1>     ; Update last modification date&time of file
15865                                   <1>     ; (call from syswrite -> writei)
15866                                   <1>     ; ((also updates file size)) // 26/10/2016
15867                                   <1>     ;
15868                                   <1>     ; INPUT:
15869                                   <1>     ;       byte [setfmod] = open file number
15870                                   <1>     ; OUTPUT:
15871                                   <1>     ;       cf = 0 -> success !
15872                                   <1>     ;       cf = 1 -> lmdt update has been failed!
15873                                   <1>     ;
15874                                   <1>     ; Modified registers: eax, ebx, ecx, edx, esi, edi
15875                                   <1>     ;
15876                                   <1>
15877                                   <1>     ;cmp    byte [setfmod], 0
15878                                   <1>     ;jna    short uflmdt_2 ; nothing to do
15879                                   <1>
15880                                   <1>     ; 27/08/2024
15881                                   <1>     ; if [ebx+OF_FCLUSTER] = 0 and [setfclust] > 0
15882                                   <1>     ; [setfclust] will be copied to Directory entry's
15883                                   <1>     ; first cluster fields (LO/HI).
15884                                   <1>     ; (and [setfclust] will be cleared.)
15885                                   <1>
15886 00011030 31C0                    <1>     xor     eax, eax
15887                                   <1>
15888 00011032 0FB61D[2F830100]        <1>     movzx   ebx, byte [setfmod]
15889 00011039 FECB                    <1>     dec     bl ; open file index number (0 based)
15890                                   <1>
15891 0001103B 8AA3[C4830100]          <1>     mov     ah, [ebx+OF_DRIVE]
15892 00011041 BE00010900              <1>     mov     esi, Logical_DOSDisks
15893 00011046 01C6                    <1>     add     esi, eax
15894 00011048 C0E302                  <1>     shl     bl, 2 ; *4
15895 0001104B 8B8B[44830100]          <1>     mov     ecx, [ebx+OF_FCLUSTER] ; first cluster
15896 00011051 8B93[C4850100]          <1>     mov     edx, [ebx+OF_DIRCLUSTER] ; dir cluster
15897                                   <1>
15898 00011057 D0EB                    <1>     shr     bl, 1 ; /2
15899 00011059 0FB7BB[C4870100]        <1>     movzx   edi, word [ebx+OF_DIRENTRY]
15900                                   <1>
15901                                   <1>     ; 03/09/2024
15902 00011060 80C441                  <1>     add     ah, 'A'
15903 00011063 8A4603                  <1>     mov     al, [esi+LD_FATType]
15904                                   <1>
15905 00011066 803D[697E0100]01        <1>     cmp     byte [DirBuff_ValidData], 1
15906 0001106D 727E                    <1>     jb      short uflmdt_4
15907                                   <1>
15908                                   <1>     ;mov    al, [DirBuff_DRV]
15909                                   <1>     ;sub    al, 'A'
15910                                   <1>     ;cmp    al, ah
15911                                   <1>     ; 03/09/2024
15912                                   <1>     ; 27/08/2024
15913                                   <1>     ;add    ah, 'A'
15914                                   <1>     ;mov    al, [esi+LD_FATType]
15915 0001106F 3A25[677E0100]          <1>     cmp     ah, [DirBuff_DRV]
15916 00011075 7576                    <1>     jne     short uflmdt_4 ; different drive
15917                                   <1>     ;mov    al, [esi+LD_FATType]
15918 00011077 3A05[687E0100]          <1>     cmp     al, [DirBuff_FATType]
15919 0001107D 756E                    <1>     jne     short uflmdt_5 ; different FS type
15920 0001107F 3B15[6E7E0100]          <1>     cmp     edx, [DirBuff_Cluster]
15921 00011085 7566                    <1>     jne     short uflmdt_5 ; different cluster
15922                                   <1>
15923                                   <1> uflmdt_1:
15924                                   <1>     ; Directory buffer is ready here!
15925                                   <1>     ; OF_FCLUSTER must be compared/verified
15926 00011087 BE00000800              <1>     mov     esi, Directory_Buffer
15927                                   <1>     ;shl    di, 5 ; dir entry index * 32
15928                                   <1>     ; 23/07/2022
15929 0001108C C1E705                  <1>     shl     edi, 5
15930 0001108F 01FE                    <1>     add     esi, edi ; offset
15931                                   <1>     ;
15932                                   <1>     ; 03/09/2024
15933                                   <1>     ;test   byte [esi+DirEntry_Attr], 18h ; Vol & Dir
15934                                   <1>     ;jnz    short uflmdt_2 ; not a valid file !
15935 00011091 668B4614                <1>     mov     ax, [esi+DirEntry_FstClusHI]
15936 00011095 C1E010                  <1>     shl     eax, 16
15937 00011098 668B461A                <1>     mov     ax, [esi+DirEntry_FstClusLO]
15938                                   <1>
15939                                   <1>     ; 03/09/2024 (short jump)
15940 0001109C F6460B18                <1>     test    byte [esi+DirEntry_Attr], 18h ; Vol & Dir
15941 000110A0 7571                    <1>     jnz     short uflmdt_2 ; not a valid file !
15942                                   <1>
15943 000110A2 39C8                    <1>     cmp     eax, ecx ; same first cluster ?
15944                                   <1>     ;je     short uflmdt_3 ; yes, it is OK !!!
15945                                   <1>     ; 23/07/2022
15946 000110A4 756D                    <1>     jne     short uflmdt_2
15947                                   <1>
15948                                   <1>     ;;;
15949                                   <1>     ; 03/09/2024
```

```
15950 000110A6 D0E3                <1>     shl     bl, 1 ; *2
15951                              <1>     ; 27/08/2024
15952 000110A8 21C0                <1>     and     eax, eax
15953 000110AA 751B                <1>     jnz     short uflmdt_3 ; not empty file
15954 000110AC 8705[30830100]      <1>     xchg    eax, [setfclust]
15955 000110B2 09C0                <1>     or      eax, eax
15956 000110B4 7411                <1>     jz      short uflmdt_3
15957                              <1>     ; 03/09/2024
15958 000110B6 8983[44830100]      <1>     mov     [ebx+OF_FCLUSTER], eax
15959                              <1>     ;
15960 000110BC 6689461A            <1>     mov     [esi+DirEntry_FstClusLO], ax
15961 000110C0 C1E810              <1>     shr     eax, 16
15962 000110C3 66894614            <1>     mov     [esi+DirEntry_FstClusHI], ax
15963                              <1>     ;;;
15964                              <1>
15965                              <1> uflmdt_3:
15966                              <1>     ; Update directory entry
15967                              <1>     ; 03/09/2024
15968                              <1>     ; 26/10/2016
15969                              <1>     ;shl    bl, 1 ; *2
15970 000110C7 8B83[C4840100]      <1>     mov     eax, [ebx+OF_SIZE] ; file size
15971 000110CD 89461C              <1>     mov     [esi+DirEntry_FileSize], eax
15972                              <1>     ;
15973 000110D0 E8179CFFFF          <1>     call    convert_current_date_time
15974                              <1>     ; OUTPUT -> DX = Date in dos dir entry format
15975                              <1>       ;      AX = Time in dos dir entry format
15976 000110D5 66894616            <1>     mov     [esi+DirEntry_WrtTime], ax
15977 000110D9 66895618            <1>     mov     [esi+DirEntry_WrtDate], dx
15978 000110DD 66895612            <1>     mov     [esi+DirEntry_LastAccDate], dx
15979 000110E1 C605[697E0100]02    <1>     mov     byte [DirBuff_ValidData], 2
15980                              <1>     ;call   save_directory_buffer
15981                              <1>     ;retn
15982                              <1>     ; 23/07/2022
15983 000110E8 E9949CFFFF          <1>     jmp     save_directory_buffer
15984                              <1>
15985                              <1> uflmdt_4:
15986                              <1>     ; Directory buffer sector read&write
15987                              <1>     ; 23/10/2016
15988                              <1>     ; 27/08/2024
15989                              <1>     ;mov    al, [esi+LD_FATType]
15990                              <1> uflmdt_5:
15991 000110ED BB[6C940100]        <1>     mov     ebx, rw_buffer ; Common r/w sector buffer addr
15992                              <1>
15993                              <1> ; 19/12/2025
15994                              <1> %if 0
15995                              <1>     and     al, al ; 0 = Singlix FS
15996                              <1>     jz      short uflmdt_11
15997                              <1> %endif
15998                              <1>
15999                              <1> ;uflmdt_12:
16000 000110F2 21D2                <1>     and     edx, edx
16001 000110F4 7527                <1>     jnz     short uflmdt_9
16002                              <1>
16003 000110F6 3C02                <1>     cmp     al, 2  ; 3 = FAT32
16004 000110F8 7720                <1>     ja      short uflmdt_8
16005                              <1>
16006 000110FA 89F8                <1>     mov     eax, edi ; directory entry index number
16007                              <1>     ;shr    ax, 4 ; 16 entries per sector
16008                              <1>     ; 23/07/2022
16009 000110FC C1E804              <1>     shr     eax, 4
16010 000110FF 034664              <1>     add     eax, [esi+LD_ROOTBegin]
16011                              <1>     ; eax = root directory sector
16012                              <1> uflmdt_6:
16013 00011102 50                  <1>     push    eax ; * ; disk sector address
16014 00011103 51                  <1>     push    ecx ; first cluster
16015 00011104 B901000000          <1>     mov     ecx, 1
16016                              <1>     ; ecx = sector count
16017 00011109 E8B4080000          <1>     call    disk_read
16018 0001110E 59                  <1>     pop     ecx
16019 0001110F 7320                <1>     jnc     short uflmdt_10
16020 00011111 58                  <1>     pop     eax ; *
16021                              <1> uflmdt_7:
16022 00011112 C3                  <1>     retn
16023                              <1>
16024                              <1> uflmdt_2:
16025                              <1>     ; save directory buffer if has modified/changed sign
16026                              <1>     ; (It is good to save dir buff even if the searched
16027                              <1>     ; directory entry is not found !?)
16028 00011113 E8699CFFFF          <1>     call    save_directory_buffer
16029 00011118 F9                  <1>     stc     ; update failed
16030 00011119 C3                  <1>     retn
16031                              <1>
16032                              <1> ; 19/12/2025
16033                              <1> %if 0
16034                              <1> uflmdt_11:
16035                              <1>     ; 24/10/2016
16036                              <1>     ; Update last modification date & time of a file
16037                              <1>     ; on a disk with Singlix File System.
16038                              <1>     ;
16039                              <1>     ; (Method: Read the FDT -File Description Table-
16040                              <1>     ; sector of the file and update the lmdt data fields,
16041                              <1>     ; then write FDT sector to the disk.
16042                              <1>     ; /// It is easy but there is compatibility buffer
16043                              <1>     ; method also for changing directory entry data and
16044                              <1>     ; also there are some programming issues for Singlix
16045                              <1>     ; file system (TRFS), which are not completed yet!)
16046                              <1>     ;
16047                              <1>     ; Not ready yet ! (24/10/2016)
16048                              <1>     ; /// Temporary code for error return ! ///
16049                              <1>     xor     eax, eax
16050                              <1>     stc
16051                              <1>     retn
16052                              <1> %endif
16053                              <1>
16054                              <1> uflmdt_8:
16055 0001111A 8B5632              <1>     mov     edx, [esi+LD_BPB+FAT32_RootFClust]
16056                              <1> uflmdt_9:
16057 0001111D 83FA02              <1>     cmp     edx, 2
16058 00011120 72F0                <1>     jb      short uflmdt_7 ; invalid, nothing to do
16059                              <1>
16060                              <1>     ;sub    edx, 2
16061                              <1>     ; 30/07/2022
16062 00011122 4A                  <1>     dec     edx
16063 00011123 4A                  <1>     dec     edx
16064 00011124 89D0                <1>     mov     eax, edx
16065 00011126 0FB65613            <1>     movzx   edx, byte [esi+LD_BPB+SecPerClust]
16066 0001112A F7E2                <1>     mul     edx
16067 0001112C 034668              <1>     add     eax, [esi+LD_DATABegin]
16068                              <1>     ; eax = sub directory (data) sector
16069 0001112F EBD1                <1>     jmp     short uflmdt_6
16070                              <1>
16071                              <1> uflmdt_10:
16072                              <1>     ; Directory sector buffer is ready here!
16073                              <1>     ; OF_FCLUSTER must be compared/verified
```

```
16074                                   <1>     ; edi = dir entry index number (<= 2047)
16075 00011131 6683E70F                 <1>     and     di, 0Fh ; 16 entries per sector
16076                                   <1>     ;shl    di, 5 ; dir entry index * 32
16077                                   <1>     ; 23/07/2022
16078 00011135 C1E705                   <1>     shl     edi, 5
16079 00011138 81C7[6C940100]           <1>     add     edi, rw_buffer
16080                                   <1>     ;
16081 0001113E F6470B18                 <1>     test    byte [edi+DirEntry_Attr], 18h ; Vol & Dir
16082 00011142 75CF                     <1>     jnz     short uflmdt_2 ; not a valid file !
16083 00011144 668B5714                 <1>     mov     dx, [edi+DirEntry_FstClusHI]
16084 00011148 C1E210                   <1>     shl     edx, 16
16085 0001114B 668B571A                 <1>     mov     dx, [edi+DirEntry_FstClusLO]
16086 0001114F 39CA                     <1>     cmp     edx, ecx ; same first cluster ?
16087 00011151 75C0                     <1>     jne     short uflmdt_2 ; no !?
16088                                   <1>
16089                                   <1>     ;;;
16090                                   <1>     ; 27/08/2024
16091 00011153 21D2                     <1>     and     edx, edx
16092 00011155 7515                     <1>     jnz     short uflmdt_12 ; not empty file
16093 00011157 8715[30830100]           <1>     xchg    edx, [setfclust]
16094 0001115D 09D2                     <1>     or      edx, edx
16095 0001115F 740B                     <1>     jz      short uflmdt_12
16096 00011161 6689571A                 <1>     mov     [edi+DirEntry_FstClusLO], dx
16097 00011165 C1EA10                   <1>     shr     edx, 16
16098 00011168 66895714                 <1>     mov     [edi+DirEntry_FstClusHI], dx
16099                                   <1> uflmdt_12:
16100                                   <1>     ;;;
16101                                   <1>
16102                                   <1>     ; Update directory entry
16103 0001116C E87B9BFFFF               <1>     call    convert_current_date_time
16104                                   <1>     ; OUTPUT -> DX = Date in dos dir entry format
16105                                   <1>     ;       AX = Time in dos dir entry format
16106 00011171 66894716                 <1>     mov     [edi+DirEntry_WrtTime], ax
16107 00011175 66895718                 <1>     mov     [edi+DirEntry_WrtDate], dx
16108 00011179 66895712                 <1>     mov     [edi+DirEntry_LastAccDate], dx
16109                                   <1>
16110 0001117D 58                       <1>     pop     eax ; *
16111                                   <1>
16112 0001117E BB[6C940100]             <1>     mov     ebx, rw_buffer ; Common r/w sector buffer addr
16113 00011183 B901000000               <1>     mov     ecx, 1
16114                                   <1>     ; esi = logical dos description table address
16115                                   <1>     ; eax = disk sector number/address (LBA)
16116                                   <1>     ; ecx = sector count
16117                                   <1>     ; ebx = buffer address
16118 00011188 E826080000               <1>     call    disk_write
16119 0001118D 7284                     <1>     jc      short uflmdt_2
16120                                   <1> ;uflmdt_12:
16121                                   <1>     ; save directory buffer if has modified/changed sign
16122                                   <1>     ;call  save_directory_buffer
16123                                   <1>     ;retn
16124                                   <1>     ; 23/07/2022
16125 0001118F E9ED9BFFFF               <1>     jmp     save_directory_buffer
16126                                   <1>
16127                                   <1> sysalloc:
16128                                   <1>     ; 23/07/2022 - TRDOS 386 v2.0.5
16129                                   <1>     ; 14/10/2017
16130                                   <1>     ; 20/08/2017 - 01/09/2017
16131                                   <1>     ; 20/02/2017 - 04/03/2017 - 15/05/2017
16132                                   <1>     ; 19/02/2017 - TRDOS 386 (TRDOS v2.0)
16133                                   <1>     ; (TRDOS 386 feature only!)
16134                                   <1>     ;
16135                                   <1>     ; Allocate Contiguous Memory Block/Pages (for user)
16136                                   <1>     ; (System call for DMA Buffer allocation etc.)
16137                                   <1>     ;
16138                                   <1>     ; INPUT ->
16139                                   <1>     ;       EBX = Virtual address (for user)
16140                                   <1>     ;            (Physical memory block/aperture
16141                                   <1>     ;            will be mapped to this virtual address)
16142                                   <1>     ;       ECX = Byte Count
16143                                   <1>     ;            (will be rounded up to page border)
16144                                   <1>     ;       If ECX = 0
16145                                   <1>     ;           System call will return with an error (cf=1)
16146                                   <1>     ;           but ECX will contain maximum size of
16147                                   <1>     ;           available memory aperture and physical
16148                                   <1>     ;           (beginning) address of that aperture
16149                                   <1>     ;           (which have maximum size) will be in EAX.
16150                                   <1>     ;       EDX = Upper limit of the requested physical memory
16151                                   <1>     ;            block/pages.
16152                                   <1>     ;           (The last byte address of the memory aperture
16153                                   <1>     ;            must not be equal to or above this limit.)
16154                                   <1>     ;       If EDX = 0
16155                                   <1>     ;          there is NOLIMIT !
16156                                   <1>     ;       If EDX = 0FFFFFFFFh (-1)
16157                                   <1>     ;          ESI = Lower Limit !
16158                                   <1>     ;               (Beginning of the block must not be 'less'
16159                                   <1>     ;               than this.) (Must be equal to or above...)
16160                                   <1>     ;          EDI = Upper Limit !
16161                                   <1>     ;               (End of the block must be !less! than this)
16162                                   <1>     ;               (The last byte addr of the memory aperture
16163                                   <1>     ;               must not be equal to or above this limit.)
16164                                   <1>     ;
16165                                   <1>     ; OUTPUT ->
16166                                   <1>     ;       If CF = 0
16167                                   <1>     ;       EAX = Physical address of the allocated memory block
16168                                   <1>     ;       ECX = Allocated bytes (as rounded up to page borders)
16169                                   <1>     ;       EBX = Virtual address (as rounded up)
16170                                   <1>     ;       IF CF = 1
16171                                   <1>     ;           Requested (size of) Memory block could not be
16172                                   <1>     ;           allocated to the user!
16173                                   <1>     ;       IF CF = 1 & EAX = 0 (Insufficient memory error!)
16174                                   <1>     ;          ECX = Total number of free bytes
16175                                   <1>     ;                (not size of available contiguous bytes!)
16176                                   <1>     ;       If CF = 1 & EAX > 0
16177                                   <1>     ;          there is not a memory aperture with requested size
16178                                   <1>     ;          but total free mem is not less than requested size.
16179                                   <1>     ;          EAX = Physical addr of available memory aperture
16180                                   <1>     ;               with max size
16181                                   <1>     ;               (but it doesn't fit to the conditions!)
16182                                   <1>     ;          ECX = Size of available memory aperture in bytes.
16183                                   <1>     ;       If CF = 1 -> EAX = 0FFFFFFFFh
16184                                   <1>     ;          Conditions/Parameters are wrong !
16185                                   <1>     ;          ECX is same with input value.
16186                                   <1>     ;
16187                                   <1>     ; Note: Previously allocated pages will be deallocated if
16188                                   <1>     ;       new allocation conditions are met.
16189                                   <1>     ;
16190                                   <1>     ; Note: u.break control may be included in future versions
16191                                   <1>     ;
16192                                   <1>
16193 00011194 31C0                     <1>     xor     eax, eax ; 0
16194                                   <1>     ; 14/10/2017
16195 00011196 4A                       <1>     dec     edx ; is there a limit ?
16196 00011197 7810                     <1>     js      short sysalloc_1 ;  0 -> 0FFFFFFFFh -> NO LIMIT
16197 00011199 42                       <1>     inc     edx ; > 0
```

```
16198                                   <1>     ; Check upper address limit
16199                                   <1>     ;(round up to page borders)
16200 0001119A 81C1FF0F0000             <1>     add     ecx, PAGE_SIZE-1 ; 4095
16201 000111A0 6681E100F0               <1>     and     cx, ~PAGE_OFF ; not 4095
16202 000111A5 39CA                     <1>     cmp     edx, ecx ; upper limit - block size
16203 000111A7 7224                     <1>     jb      short sysalloc_err
16204                                   <1> sysalloc_1:
16205                                   <1>     ; EAX = Beginning address (physical)
16206                                   <1>     ; EAX = 0 -> Allocate mem block from the 1st proper aperture
16207                                   <1>     ; ECX = Number of bytes to be allocated
16208 000111A9 E88B4BFFFF              <1>     call    allocate_memory_block
16209 000111AE 721D                     <1>     jc      short sysalloc_err
16210                                   <1>     ; 01/09/2017
16211 000111B0 29C2                     <1>     sub     edx, eax ; upper limit address - beginning address
16212 000111B2 760F                     <1>     jna     short sysalloc_3 ; begin addr not less than the limit
16213 000111B4 39CA                     <1>     cmp     edx, ecx
16214 000111B6 720B                     <1>     jb      short sysalloc_3 ; end address overs the limit
16215                                   <1> sysalloc_2:
16216                                   <1>     ; EAX = Beginning (physical) addr of the allocated mem block
16217                                   <1>     ; ECX = Num of allocated bytes (rounded up to page borders)
16218 000111B8 50                       <1>     push    eax ; * ; 04/03/2017
16219                                   <1>     ; Here, requested contiquous memory pages have been allocated
16220                                   <1>     ; on Memory Allocation Table but user's page directory
16221                                   <1>     ; and page tables have not been updated yet!
16222 000111B9 51                       <1>     push    ecx ; **
16223                                   <1>     ; ebx = virtual address (will be rounded up to page border)
16224                                   <1>     ; ecx = number of bytes to be deallocated
16225                                   <1>     ;       will be adjusted to ebx+ecx round down - ebx round up
16226 000111BA E8D24EFFFF              <1>     call    deallocate_user_pages
16227 000111BF 731F                     <1>     jnc     short sysalloc_4 ; EAX = Deallocated memory bytes
16228 000111C1 59                       <1>     pop     ecx ; **
16229 000111C2 58                       <1>     pop     eax ; *
16230                                   <1> sysalloc_3:
16231                                   <1>     ; error !
16232                                   <1>     ; restore Memory Allocation Table Content
16233 000111C3 E88B4DFFFF              <1>     call    deallocate_memory_block
16234 000111C8 31C0                     <1>     xor     eax, eax ; 0
16235 000111CA 48                       <1>     dec     eax ; 0FFFFFFFFh ; 15/05/2017
16236 000111CB EB09                     <1>     jmp     short sysalloc_wrong
16237                                   <1> sysalloc_err:
16238 000111CD 8B2D[308E0100]           <1>     mov     ebp, [u.usp]  ; ebp points to user's registers
16239 000111D3 894D18                   <1>     mov     [ebp+24], ecx ; return to user with ecx value
16240                                   <1> sysalloc_wrong:
16241                                   <1>     ; eax = 0FFFFFFFFh
16242 000111D6 A3[348E0100]             <1>     mov     [u.r0], eax
16243 000111DB E97EB8FFFF              <1>     jmp     error
16244                                   <1> sysalloc_4:
16245 000111E0 8B2D[308E0100]           <1>     mov     ebp, [u.usp]  ; ebp points to user's registers
16246 000111E6 894518                   <1>     mov     [ebp+24], eax ; return to user with ecx value
16247 000111E9 895D10                   <1>     mov     [ebp+16], ebx ; new value of ebx (rounded up)
16248 000111EC 89C1                     <1>     mov     ecx, eax ; byte count (from 'deallocate_user_pages')
16249 000111EE 5A                       <1>     pop     edx ; ** ; discard (another) byte count
16250 000111EF 58                       <1>     pop     eax ; *
16251 000111F0 A3[348E0100]             <1>     mov     [u.r0], eax ; physical address
16252                                   <1>
16253 000111F5 51                       <1>     push    ecx ; 20/08/2017
16254                                   <1>     ;
16255                                   <1>     ; Write newly allocated contiguous (physical) pages
16256                                   <1>     ; on page dir and page tables of current user/process
16257                                   <1>     ; as PRESENT, USER, WRITABLE
16258                                   <1>     ; (then clear allocated pages)
16259 000111F6 E87E4FFFFF              <1>     call    allocate_user_pages
16260                                   <1>     ;jnc    sysret ; OK! return to process with success...
16261                                   <1>
16262                                   <1>     ; 20/08/2017 ('sysdma' modification)
16263 000111FB 59                       <1>     pop     ecx
16264 000111FC A1[348E0100]             <1>     mov     eax, [u.r0]   ; physical address (of the block)
16265                                   <1>
16266 00011201 7219                     <1>     jc      short sysalloc_6
16267                                   <1>
16268 00011203 833D[F88C0100]FF         <1>     cmp     dword [dma_addr], 0FFFFFFFFh ; -1
16269                                   <1>     ;jb     sysret
16270                                   <1>     ; 23/07/2022
16271 0001120A 720B                     <1>     jb      short sysalloc_5 ; jmp sysret
16272                                   <1>
16273 0001120C A3[F88C0100]             <1>     mov     [dma_addr], eax ; save dma address for sysdma
16274 00011211 890D[FC8C0100]           <1>     mov     [dma_size], ecx ; save dma buff size for sysdma
16275                                   <1> sysalloc_5:
16276 00011217 E962B8FFFF              <1>     jmp     sysret
16277                                   <1>
16278                                   <1> sysalloc_6:
16279                                   <1>     ;
16280                                   <1>     ; unexpected error ! insufficient memory !? conflict !?
16281                                   <1>     ; (!!?there is not a free page for a new page table?!!)
16282                                   <1>     ; We need to terminate process with error message !!!
16283                                   <1>     ;
16284 0001121C 8B2D[308E0100]           <1>     mov     ebp, [u.usp]  ; ebp points to user's registers
16285 00011222 8B4D18                   <1>     mov     ecx, [ebp+24] ; byte count
16286                                   <1>
16287                                   <1>     ; 20/08/2017
16288                                   <1>     ;mov    eax, [u.r0]   ; physical address (of the block)
16289                                   <1>
16290                                   <1>     ;
16291                                   <1>     ; restore Memory Allocation Table Content
16292 00011225 E8294DFFFF              <1>     call    deallocate_memory_block
16293                                   <1>     ;
16294 0001122A 803D[1E680000]03         <1>     cmp     byte [CRT_MODE], 3 ; 80x25 text mode?
16295 00011231 7407                     <1>     je      short sysalloc_7 ; yes
16296                                   <1>     ; Current mode is VGA (or CGA graphics) mode,
16297                                   <1>     ; We need to return to text mode for displaying
16298                                   <1>     ; error message just before 'sysexit'.
16299 00011233 B003                     <1>     mov     al, 3
16300 00011235 E84F09FFFF              <1>     call    _set_mode
16301                                   <1> sysalloc_7:
16302 0001123A BE[C7310100]             <1>     mov     esi, beep_Insufficient_Memory ; error message
16303 0001123F E8665BFFFF              <1>     call    print_msg ; print/display the message
16304 00011244 B801000000              <1>     mov     eax, 1 ; ax=1 is needed for 'sysexit' procedure
16305 00011249 E96EB9FFFF              <1>     jmp     sysexit ; and terminate the process !
16306                                   <1>
16307                                   <1> sysdalloc:
16308                                   <1>     ; 19/02/2017 - TRDOS 386 (TRDOS v2.0)
16309                                   <1>     ; (TRDOS 386 feature only!)
16310                                   <1>     ;
16311                                   <1>     ; Deallocate Memory Block/Pages (for user)
16312                                   <1>     ; (Complementary call for sysalloc.)
16313                                   <1>     ;
16314                                   <1>     ; INPUT ->
16315                                   <1>     ;       EBX = Virtual address (for user)
16316                                   <1>     ;             (will be rounded up to page border)
16317                                   <1>     ;       ECX = Byte Count
16318                                   <1>     ;            (will be adjusted to page borders)
16319                                   <1>     ;       If ICX = 0
16320                                   <1>     ;          nothing to do
16321                                   <1>     ;       If EBX + ECX > User's ESP
```

```
16322                                  <1>     ;         nothing to do
16323                                  <1>     ;
16324                                  <1>     ; Note: u.break control may be included in future versions
16325                                  <1>     ;
16326                                  <1>     ; OUTPUT ->
16327                                  <1>     ;      If CF = 0
16328                                  <1>     ;         EAX = Deallocated memory bytes
16329                                  <1>     ;         EBX = Virtual address (as rounded up)
16330                                  <1>     ;      IF CF = 1
16331                                  <1>     ;         EAX = 0
16332                                  <1>     ;
16333                                  <1>     ; Note: Main purpose of this call is to deallocate/release
16334                                  <1>     ;      previously allocated (physically) contiguous memory
16335                                  <1>     ;      pages but beginning (virtual) address may not be
16336                                  <1>     ;      followed by physically contiguous pages. So, this
16337                                  <1>     ;      system call will deallocate user's virtually
16338                                  <1>     ;      contiguous memory pages. Also, there is not any
16339                                  <1>     ;      objections to use this system call without sysalloc
16340                                  <1>     ;      system call; only possible objection is to lost data
16341                                  <1>     ;      within user's memory space, if the beginning address
16342                                  <1>     ;      and size is not proper.
16343                                  <1>     ;
16344                                  <1>     ; Note: Empty page tables will not be deallocated!!!
16345                                  <1>     ;       (they will be deallocated at process termination)
16346                                  <1>     ;
16347                                  <1>     ; Note: When the program terminates itself or when it is
16348                                  <1>     ;      terminated by operating system kernel, all allocated
16349                                  <1>     ;      memory pages will be deallocated during termination
16350                                  <1>     ;      stage. So, 'sysdalloc' is not necessary except
16351                                  <1>     ;      forgiving memory block to other programs/processes.
16352                                  <1>     ;
16353 0001124E 8B15[2C8E0100]           <1>     mov     edx, [u.sp]
16354 00011254 8B420C                 <1>     mov     eax, [edx+12] ; user's stack pointer
16355 00011257 29C8                   <1>     sub     eax, ecx ; esp - byte count
16356 00011259 24FC                   <1>     and     al, 0FCh ; dword alignment
16357 0001125B 39D8                   <1>     cmp     eax, ebx
16358 0001125D 7220                   <1>     jb      short sysdalloc_err ; deallocation overlaps with stack
16359                                  <1>
16360 0001125F 31C0                   <1>     xor     eax, eax
16361 00011261 21C9                   <1>     and     ecx, ecx
16362 00011263 7407                   <1>     jz      short sysdalloc_2
16363                                  <1>
16364 00011265 E8274EFFFF             <1>     call    deallocate_user_pages
16365 0001126A 7213                   <1>     jc      short sysdalloc_err
16366                                  <1>
16367                                  <1> sysdalloc_2:
16368 0001126C A3[348E0100]           <1>     mov     [u.r0], eax
16369 00011271 8B2D[308E0100]         <1>     mov     ebp, [u.usp]
16370 00011277 895D10                 <1>     mov     [ebp+16], ebx ; new value of ebx
16371 0001127A E9FFB7FFFF             <1>     jmp     sysret
16372                                  <1>
16373                                  <1> sysdalloc_err:
16374 0001127F A3[348E0100]           <1>     mov     [u.r0], eax ; 0
16375 00011284 E9D5B7FFFF             <1>     jmp     error
16376                                  <1>
16377                                  <1> syscalbac:
16378                                  <1>     ; SYS CALLBACK
16379                                  <1>     ; 23/07/2022 - TRDOS 386 v2.0.5
16380                                  <1>     ; 03/08/2020
16381                                  <1>     ; 16/04/2017
16382                                  <1>     ; 14/04/2017
16383                                  <1>     ; 13/04/2017
16384                                  <1>     ; 28/02/2017
16385                                  <1>     ; 26/02/2017
16386                                  <1>     ; 24/02/2017
16387                                  <1>     ; 21/02/2017 - TRDOS 386 (TRDOS v2.0)
16388                                  <1>     ; (TRDOS 386 feature only!)
16389                                  <1>     ;
16390                                  <1>     ; Link or unlink IRQ callback service to/from user (ring 3)
16391                                  <1>     ;
16392                                  <1>     ; INPUT ->
16393                                  <1>     ;      BL = IRQ number (Hardware interrupt request number)
16394                                  <1>     ;           (0 t0 15 but IRQ 0,1,2,6,8,14,15 are prohibited)
16395                                  <1>     ;           IRQ numbers 3,4,5,7,9,10,11,12,13 are valid
16396                                  <1>     ;           (numbers >15 are invalid)
16397                                  <1>     ;
16398                                  <1>     ;      BH = 0 = Unlink IRQ (in BL) from user (ring 3) service
16399                                  <1>     ;           1 = Link IRQ by using Signal Response Byte method
16400                                  <1>     ;           2 = Link IRQ by using Callback service method
16401                                  <1>     ;           3 = Link IRQ by using Auto Increment S.R.B. method
16402                                  <1>     ;          >3 = invalid
16403                                  <1>     ;
16404                                  <1>     ;      CL = Signal Return/Response Byte value
16405                                  <1>     ;
16406                                  <1>     ;      If BH = 3, kernel will put a counter value  ; 03/08/2020
16407                                  <1>     ;                (into the S.R.B. addr)
16408                                  <1>     ;                between 0 to 255. (start value = CL+1)
16409                                  <1>     ;
16410                                  <1>     ;      NOTE: counter value, for example: even and odd numbers
16411                                  <1>     ;            may be used for -audio- DMA buffer switch
16412                                  <1>     ;            within double buffer method, etc.
16413                                  <1>     ;
16414                                  <1>     ;      EDX = Signal return (Response) byte address
16415                                  <1>     ;                  - or -
16416                                  <1>     ;            Interrupt/Callback service/routine address
16417                                  <1>     ;
16418                                  <1>     ;            (virtual address in user's memory space)
16419                                  <1>     ;
16420                                  <1>     ; OUTPUT ->
16421                                  <1>     ;      CF = 0 & EAX = 0 -> Successful setting
16422                                  <1>     ;      CF = 1 & EAX > 0 -> IRQ is prohibited or locked
16423                                  <1>     ;                          by another process
16424                                  <1>     ;                eax = ERR_PERM_DENIED -> prohibited or locked
16425                                  <1>     ;                eax = ERR_INV_PARAMETER ->
16426                                  <1>     ;                      invalid parameter/option or bad address
16427                                  <1>     ;
16428                                  <1>     ;      NOTE: Timer callbacks are set by using 'systimer'
16429                                  <1>     ;            system call (IRQ 0, PIT and IRQ 8, RTC)
16430                                  <1>     ;
16431                                  <1>     ;            Direct keyboard access is performed by using
16432                                  <1>     ;            Keyboard Interrupt (INT 32h)
16433                                  <1>     ;
16434                                  <1>     ;            It is prohibited here because:
16435                                  <1>     ;             1) Signal Response Byte method has not advantage
16436                                  <1>     ;                against INT 32h, function AH = 1. Also,
16437                                  <1>     ;                keyboard service interrupt will return with
16438                                  <1>     ;                ascii and scan codes (AL, AH) while
16439                                  <1>     ;                SRB method has only 1 byte space for ascii code
16440                                  <1>     ;                or scan code. One byte signal response is used
16441                                  <1>     ;                for ensuring very simple and very fast
16442                                  <1>     ;                virtual to physical memory address conversion
16443                                  <1>     ;                without any memory page crossover risk.
16444                                  <1>     ;                (Otherwise double page conversion or word
16445                                  <1>     ;                alignment would be needed.)
```

```
16446                                 <1>     ;             2) Badly written user code (callback code)
16447                                 <1>     ;                can prevent keyboard and timesharing functions
16448                                 <1>     ;                of the operating system via continuous and long
16449                                 <1>     ;                keyboard event handling by callback service.
16450                                 <1>     ;                (It can cause to lose immediate keystroke
16451                                 <1>     ;                response from hardware to user.)
16452                                 <1>     ;             3) If user will check any keyboard events, 'getkey'
16453                                 <1>     ;                (or 'getchar') must have more priority than other
16454                                 <1>     ;                (video etc.) events because only control ability
16455                                 <1>     ;                on a procedural infinite loop is a keyboard or
16456                                 <1>     ;                mouse event. So user can use keyboard function
16457                                 <1>     ;                at the end or at the beginning of a loop.
16458                                 <1>     ;                In this case, INT 32h is used for that purpose
16459                                 <1>     ;                and timer interrupt etc. callbacks can be used
16460                                 <1>     ;                for dynamic and synchronized data refresh/transfer
16461                                 <1>     ;                while cpu is in a static loop (without polling).
16462                                 <1>     ;                Keyboard Int callback is not more useful because
16463                                 <1>     ;                already a manual check (a key is pressed or not)
16464                                 <1>     ;                can be performed (via INT 32h, AH = 1) efficiently
16465                                 <1>     ;                in a loop to prevent a locked infinitive loop.
16466                                 <1>     ;
16467                                 <1>     ;          Disk IRQs (6,14,15) have been phohibited from ring 3
16468                                 <1>     ;          callback because, disk operations (file system services
16469                                 <1>     ;          etc.) are independent from user program, for fast disk r/w.
16470                                 <1>     ;          They are not more useful at ring 3 while they are in use
16471                                 <1>     ;          by standard diskio functions which are mandatory part of
16472                                 <1>     ;          (monolithic) OS kernel and mainprog command interpreter.
16473                                 <1>     ;          INT 33h diskio functions are enough for user level disk
16474                                 <1>     ;          r/w.
16475                                 <1>     ;
16476                                 <1>     ; TRDOS 386 - IRQ CALLBACK structures (parameters):
16477                                 <1>     ;
16478                                 <1>     ;         [u.irqlock] = 1 word, IRQ flags (0-15) that indicates
16479                                 <1>     ;                     which IRQs are locked by (that) user.
16480                                 <1>     ;                      Lock and unlock (by user) will change
16481                                 <1>     ;                     these flags or 'terminate process' (sysexit)
16482                                 <1>     ;                     will clear these flags and unlock those IRQs.
16483                                 <1>     ;
16484                                 <1>     ;                     Bit 0 is for IRQ 0 and Bit 15 is for IRQ 15
16485                                 <1>     ;
16486                                 <1>     ;          IRQ(x).owner        : 1 byte, user, [u.uno], 0 = free (unlocked)
16487                                 <1>     ;
16488                                 <1>     ;          IRQ(x).method : 1 byte for callback method & status
16489                                 <1>     ;                          0 = Signal Response Byte method
16490                                 <1>     ;                          1 = Callback service method
16491                                 <1>     ;                          >1 = invalid for current 'syscalback'.
16492                                 <1>     ;                        or(+) 80h = IRQ is in use by system (ring 0)
16493                                 <1>     ;                                    function (audio etc.) or
16494                                 <1>     ;                                   a device driver.
16495                                 <1>     ;                        (system function will ignore the lock/owner)
16496                                 <1>     ;
16497                                 <1>     ;          IRQ(x).srb  : 1 byte, Signal Return/Response byte value
16498                                 <1>     ;                         (a fixed value by user or a counter value
16499                                 <1>     ;                         from 0 to 255, which is increased by every
16500                                 <1>     ;                         interrupt just before putting it into
16501                                 <1>     ;                         the Signal Response byte address
16502                                 <1>     ;                         (This is not used in callback serv method)
16503                                 <1>     ;
16504                                 <1>     ;          IRQ(x).addr : 1 dword
16505                                 <1>     ;                         Signal Response Byte address (physical)
16506                                 <1>     ;                                     -or-
16507                                 <1>     ;                         Callback service address (virtual)
16508                                 <1>     ;
16509                                 <1>     ;          IRQ(x).dev  : 1 byte
16510                                 <1>     ;                         0 = Default device or kernel function
16511                                 <1>     ;                                     -or-
16512                                 <1>     ;                         1-255 = Assigned device driver number
16513                                 <1>     ;
16514                                 <1>     ;          (x) = 3,4,5,7,9,10,11,12,13
16515                                 <1>     ;
16516                                 <1>     ;
16517                                 <1>     ;       NOTE: If user's process/program calls the kernel (INT 40h)
16518                                 <1>     ;             while it is already running in a (ring 3) callback
16519                                 <1>     ;             service, kernel will force (convert) system call to
16520                                 <1>     ;             'sysrele' (sys release). So, this feature provides
16521                                 <1>     ;             easy and simple usage of callback services without
16522                                 <1>     ;             falling into deepless <please 'callback me' then
16523                                 <1>     ;             let me 'callback you'> cycles! (User must return
16524                                 <1>     ;             from callback service by using 'sysrele' system
16525                                 <1>     ;             call, without a significant delay. Otherwise user
16526                                 <1>     ;             process/program may be late to catch the next event
16527                                 <1>     ;             within same callback purpose.
16528                                 <1>     ;
16529                                 <1>
16530 00011289 30C0                  <1>     xor     al, al ; the caller is 'syscalbac' sign/flag
16531 0001128B E8C1150000            <1>     call    set_irq_callback_service
16532                                 <1>     ; 16/04/2017
16533 00011290 A3[348E0100]          <1>     mov     [u.r0], eax
16534                                 <1>     ;jnc    sysret
16535                                 <1>     ; 23/07/2022
16536 00011295 7205                  <1>     jc      short syscalbac_err
16537 00011297 E9E2B7FFFF            <1>     jmp     sysret
16538                                 <1> syscalbac_err:
16539 0001129C A3[A08E0100]          <1>     mov     dword [u.error], eax
16540 000112A1 E9B8B7FFFF            <1>     jmp     error
16541                                 <1>
16542                                 <1> sysfpstat:
16543                                 <1>     ; 28/02/2017 - TRDOS 386 (TRDOS v2.0)
16544                                 <1>     ; (TRDOS 386 feature only!)
16545                                 <1>     ;
16546                                 <1>     ; Set or reset FPU registers save/restore option (for user)
16547                                 <1>     ;             (during software task switching, wswap-rswap)
16548                                 <1>     ;
16549                                 <1>     ; INPUT ->
16550                                 <1>     ;       BL = 0 -> reset
16551                                 <1>     ;       BL = 1 -> set (FPU register will be saved and restored)
16552                                 <1>     ;
16553                                 <1>     ; OUTPUT ->
16554                                 <1>     ;       cf = 0 -> no error, FPU is ready...
16555                                 <1>     ;                 (EAX = 0)
16556                                 <1>     ;       Cf = 1 -> error, 80387 FPU is not ready !
16557                                 <1>     ;                 (EAX = 0FFFFFFFFh)
16558                                 <1>
16559 000112A6 31C0                  <1>     xor     eax, eax
16560 000112A8 803D[40830100]00      <1>     cmp     byte [fpready], 0
16561 000112AF 7613                  <1>     jna     short sysfpstat_err
16562                                 <1>
16563 000112B1 80E301                <1>     and     bl, 1 ; use BIT 0 only !
16564 000112B4 881D[B38E0100]        <1>     mov     [u.fpsave], bl
16565 000112BA A3[348E0100]          <1>     mov     [u.r0], eax ; 0
16566 000112BF E9BAB7FFFF            <1>     jmp     sysret
16567                                 <1>
16568                                 <1> sysfpstat_err:
16569 000112C4 48                    <1>     dec     eax ; 0FFFFFFFFh
```

```
16570 000112C5 A3[348E0100]        <1>     mov     [u.r0], eax ; -1
16571 000112CA E98FB7FFFF          <1>     jmp     error
16572                              <1>
16573                              <1> sysdelete: ; Delete (Remove, Unlink) File
16574                              <1>     ; 29/12/2017 (TRDOS 386 = TRDOS v2.0)
16575                              <1>     ;
16576                              <1>         ; INPUT ->
16577                              <1>         ;          EBX = File name (ASCIIZ string) address
16578                              <1>     ; OUTPUT ->
16579                              <1>     ;          cf = 0 -> eax = 0
16580                              <1>     ;          cf = 1 -> Error code in AL
16581                              <1>     ;
16582                              <1>     ; Modified Registers: EAX (at the return of system call)
16583                              <1>     ;
16584                              <1>
16585 000112CF 89DE               <1>     mov     esi, ebx
16586                              <1>     ; file name is forced, change directory as temporary
16587                              <1>     ;mov    ax, 1
16588                              <1>     ;mov    [FFF_Valid], ah ; 0 ; reset
16589                              <1>     ;call   set_working_path
16590 000112D1 E8CE090000         <1>     call    set_working_path_x
16591 000112D6 731D               <1>     jnc     short sysdelete_1
16592                              <1>
16593 000112D8 21C0               <1>     and     eax, eax  ; 0 -> Bad Path!
16594 000112DA 7505               <1>     jnz     short sysdelete_err
16595                              <1>     ; eax = 0
16596                              <1> sysdelete_path_err:
16597 000112DC B813000000         <1>     mov     eax, ERR_INV_PATH_NAME ; 'bad path name !'
16598                              <1> sysdelete_err:
16599 000112E1 A3[348E0100]        <1>     mov     [u.r0], eax
16600 000112E6 A3[A08E0100]        <1>     mov     [u.error], eax
16601 000112EB E8890A0000          <1>     call    reset_working_path
16602 000112F0 E969B7FFFF          <1>     jmp     error
16603                              <1> sysdelete_1:
16604                              <1>     ;mov    esi, FindFile_Name
16605 000112F5 66B80018            <1>     mov     ax, 1800h ; Only files
16606 000112F9 E86C76FFFF          <1>     call    find_first_file
16607 000112FE 72E1               <1>     jc      short sysdelete_err
16608                              <1> sysdelete_2:
16609                              <1>     ; check file attributes
16610                              <1>
16611                              <1>     ;test   bl, 17 ; system, hidden, readonly, directory
16612 00011300 F6C307             <1>         test       bl, 7 ; system, hidden, readonly
16613 00011303 7407               <1>         jz  short sysdelete_3
16614                              <1>
16615 00011305 B80B000000         <1>         mov eax, ERR_FILE_ACCESS ; 11 = 'permission denied !'
16616 0001130A EBD5               <1>         jmp short sysdelete_err
16617                              <1> sysdelete_3:
16618 0001130C 6621D2             <1>     and     dx, dx ; Ambiguous filename chars used sign (DX>0)
16619 0001130F 7407               <1>     jz      short sysdelete_4
16620 00011311 B81A000000         <1>     mov     eax, ERR_INV_FILE_NAME ; 26 = 'invalid file name !'
16621 00011316 EBC9               <1>         jmp short sysdelete_err
16622                              <1> sysdelete_4:
16623                              <1>     ;mov    bh, [LongName_EntryLength]
16624 00011318 883D[AA800100]      <1>     mov     [DelFile_LNEL], bh ; Long name entry length (if > 0)
16625                              <1>     ; edi = Directory Entry Offset (DirBuff)
16626                              <1>     ; esi = Directory Entry (FFF Structure)
16627 0001131E E8769DFFFF         <1>     call    remove_file
16628 00011323 72BC               <1>     jc      short sysdelete_err
16629                              <1> sysrmdir_5:
16630 00011325 31C0               <1>     xor     eax, eax ; 0
16631 00011327 A3[348E0100]        <1>     mov     [u.r0], eax
16632                              <1>     ;mov    [u.error], eax
16633 0001132C E8480A0000          <1>     call    reset_working_path
16634 00011331 E948B7FFFF          <1>     jmp     sysret
16635                              <1>
16636                              <1> sysrmdir: ; Remove (Unlink) Directory
16637                              <1>     ; 17/07/2025 - TRDOS 386 v2.0.10
16638                              <1>     ; 23/07/2022 - TRDOS 386 v2.0.5
16639                              <1>     ; 19/01/2021
16640                              <1>     ; 29/12/2017 (TRDOS 386 = TRDOS v2.0)
16641                              <1>     ;
16642                              <1>         ; INPUT ->
16643                              <1>         ;          EBX = Pointer to directory name
16644                              <1>     ; OUTPUT ->
16645                              <1>     ;          cf = 0 -> eax = 0
16646                              <1>     ;          cf = 1 -> Error code in AL
16647                              <1>     ;
16648                              <1>     ; Modified Registers: EAX (at the return of system call)
16649                              <1>     ;
16650                              <1>
16651                              <1>     ; 19/01/2021
16652 00011336 803D[868E0100]00    <1>     cmp     byte [u.uid], 0  ; root (super user) ?
16653 0001133D 7614               <1>     jna     short sysrmdir_0
16654                              <1>
16655                              <1>     ;mov    dword [u.r0], ERR_PERM_DENIED
16656 0001133F B80B000000         <1>     mov     eax, ERR_PERM_DENIED ; ERR_NOT_SUPERUSER
16657 00011344 A3[348E0100]        <1>     mov     [u.r0], eax
16658 00011349 A3[A08E0100]        <1>     mov     [u.error], eax
16659 0001134E E90BB7FFFF          <1>     jmp     error
16660                              <1>
16661                              <1> sysrmdir_0:
16662 00011353 89DE               <1>     mov     esi, ebx
16663                              <1>
16664                              <1>     ;;;;
16665                              <1>     ; 17/07/2025 - TRDOS 386 v2.0.10
16666 00011355 A0[42770100]        <1>     mov     al, [Current_Drv]
16667 0001135A A2[E8800100]        <1>     mov     [rmdir_drv], al
16668 0001135F A1[3C770100]        <1>     mov     eax, [Current_Dir_FCluster]
16669 00011364 A3[E4800100]        <1>     mov     [rmdir_dir_fcluster], eax
16670                              <1>     ;;;;
16671                              <1>
16672                              <1>     ; file name is forced, change directory as temporary
16673                              <1>     ;mov    ax, 1
16674                              <1>     ;mov    [FFF_Valid], ah ; 0 ; reset
16675                              <1>     ;call   set_working_path
16676 00011369 E836090000         <1>     call    set_working_path_x
16677 0001136E 7308               <1>     jnc     short sysrmdir_1
16678                              <1>
16679 00011370 21C0               <1>     and     eax, eax  ; 0 -> Bad Path!
16680 00011372 7513               <1>     jnz     short sysrmdir_err
16681                              <1>     ; eax = 0
16682                              <1> sysrmdir_not_found:
16683                              <1>     ;mov    eax, ERR_DIR_NOT_FOUND ; Directory not found !
16684                              <1>     ; 23/07/2022
16685 00011374 B00C               <1>     mov     al, ERR_DIR_NOT_FOUND
16686 00011376 EB0F               <1>     jmp     short sysrmdir_err
16687                              <1> ;sysrmdir_err:
16688                              <1> ;   mov     [u.r0], eax
16689                              <1> ;   mov     [u.error], eax
16690                              <1> ;   call    reset_working_path
16691                              <1> ;   jmp     error
16692                              <1> sysrmdir_1:
16693                              <1>     ;mov    esi, FindFile_Name
```

```
16694 00011378 66B81008            <1>     mov     ax, 0810h ; Only directories
16695 0001137C E8E975FFFF          <1>     call    find_first_file
16696 00011381 7318                <1>     jnc     short sysrmdir_2
16697                              <1>
16698                              <1>     ; eax = 2 (File not found !)
16699 00011383 3C02                <1>     cmp     al, 2 ; ERR_NOT_FOUND
16700 00011385 74ED                <1>     je      short sysrmdir_not_found
16701                              <1>     ;jmp    short sysrmdir_err
16702                              <1>     ; 23/07/2022
16703                              <1> sysrmdir_err:
16704 00011387 A3[348E0100]        <1>     mov     [u.r0], eax
16705 0001138C A3[A08E0100]        <1>     mov     [u.error], eax
16706                              <1> sysrmdir_8: ; 23/07/2022
16707 00011391 E8E3090000          <1>     call    reset_working_path
16708 00011396 E9C3B6FFFF          <1>     jmp     error
16709                              <1> sysrmdir_2:
16710                              <1>     ; check directory attributes
16711                              <1>
16712 0001139B F6C307              <1>         test       bl, 7 ; system, hidden, readonly
16713 0001139E 7407                <1>         jz  short sysrmdir_3
16714                              <1>
16715 000113A0 B80B000000          <1>         mov eax, ERR_DIR_ACCESS ; 11 = 'permission denied !'
16716 000113A5 EBE0                <1>         jmp short sysrmdir_err
16717                              <1> sysrmdir_3:
16718 000113A7 6621D2              <1>     and     dx, dx ; Ambiguous filename chars used sign (DX>0)
16719 000113AA 7407                <1>     jz      short sysrmdir_4
16720                              <1>     ;mov    eax, ERR_NOT_DIR ; 'not a valid directory !'
16721 000113AC B813000000          <1>     mov     eax, ERR_INV_PATH_NAME ; 'bad path name !'
16722 000113B1 EBD4                <1>         jmp short sysrmdir_err
16723                              <1> sysrmdir_4:
16724                              <1>     ;mov    bh, [LongName_EntryLength]
16725 000113B3 883D[AA800100]      <1>     mov     [DelFile_LNEL], bh ; Long name entry length (if > 0)
16726                              <1>     ; edi = Directory Entry Offset (DirBuff)
16727                              <1>     ; esi = Directory Entry (FFF Structure)
16728 000113B9 E8187CFFFF          <1>     call    delete_sub_directory
16729                              <1>     ;jnc    sysrmdir_5
16730                              <1>     ; 23/07/2022
16731 000113BE 7205                <1>     jc      short sysrmdir_6
16732 000113C0 E960FFFFFF          <1>     jmp     sysrmdir_5
16733                              <1>
16734                              <1> ;   jc      short sysrmdir_6
16735                              <1> ;
16736                              <1> ;   xor     eax, eax ; 0
16737                              <1> ;sysrmdir_5:
16738                              <1> ;   mov     [u.r0], eax
16739                              <1> ;   ;mov    [u.error], eax
16740                              <1> ;   call    reset_working_path
16741                              <1> ;   jmp     sysret
16742                              <1>
16743                              <1> sysrmdir_6:
16744 000113C5 A3[348E0100]        <1>     mov     [u.r0], eax
16745 000113CA A3[A08E0100]        <1>     mov     [u.error], eax
16746                              <1>
16747 000113CF 09C0                <1>     or      eax, eax ; EAX = 0 -> Directory not empty!
16748 000113D1 7414                <1>     jz      short sysrmdir_9
16749                              <1>
16750                              <1>     ; EAX > 0 -> Error code in AL (or AX or EAX)
16751                              <1>
16752 000113D3 833D[5E7E0100]01    <1>     cmp     dword [FAT_ClusterCounter], 1
16753 000113DA 72B5                <1>     jb      short sysrmdir_8
16754                              <1> sysrmdir_7:
16755                              <1>     ; ESI = Logical DOS Drive Description Table address
16756 000113DC 66BB00FF            <1>     mov     bx, 0FF00h ; BH = FFh -> use ESI for Drive parameters
16757                              <1>                        ; BL = 0 -> Recalculate free cluster count
16758 000113E0 E87CB1FFFF          <1>     call    calculate_fat_freespace
16759                              <1>     ; 23/07/2022
16760 000113E5 EBAA                <1>     jmp     short sysrmdir_8
16761                              <1> ;sysrmdir_8:
16762                              <1> ;   call    reset_working_path
16763                              <1> ;   jmp     error
16764                              <1>
16765                              <1> sysrmdir_9:
16766 000113E7 A1[5E7E0100]        <1>     mov     eax, [FAT_ClusterCounter]
16767 000113EC 09C0                <1>     or      eax, eax ; 0 ?
16768                              <1>     ;jz     short sysrmdir_err
16769                              <1>     ; 23/07/2022
16770 000113EE 74A1                <1>     jz      short sysrmdir_8
16771                              <1>
16772                              <1>     ; ESI = Logical DOS Drive Description Table address
16773 000113F0 66BB01FF            <1>     mov     bx, 0FF01h ; BH = FFh -> use ESI for Drive parameters
16774                              <1>                        ; BL = 1 -> add free clusters
16775 000113F4 E868B1FFFF          <1>     call    calculate_fat_freespace
16776 000113F9 09C9                <1>     or      ecx, ecx
16777 000113FB 7494                <1>         jz  short sysrmdir_8 ; ecx = 0 -> OK
16778                              <1>     ; ecx > 0 -> Error (Recalculation is needed)
16779 000113FD EBDD                <1>     jmp     short sysrmdir_7
16780                              <1>
16781                              <1> syschdir: ; Change Current (Working) Drive & Directory (for user)
16782                              <1>     ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
16783                              <1>     ;
16784                              <1>         ; INPUT ->
16785                              <1>         ;           EBX = Directory name (ASCIIZ string) address
16786                              <1>     ; OUTPUT ->
16787                              <1>     ;          cf = 0 -> eax = 0
16788                              <1>     ;          cf = 1 -> Error code in AL
16789                              <1>     ;
16790                              <1>     ; Modified Registers: EAX (at the return of system call)
16791                              <1>     ;
16792                              <1>     ; NOTE: If drive name is not included, only the working
16793                              <1>     ; directory (for user, not for drive/OS) will be chanded.
16794                              <1>     ; If there is a drive name (as A:, B:, C:, D: etc.)
16795                              <1>     ; at the beginning of the ASCIIZ (directory) string,
16796                              <1>     ; working drive and working directory (for user)
16797                              <1>     ; will be changed together.
16798                              <1>     ; (When the program is terminated, MainProg -internal
16799                              <1>     ; shell- will reset working directory to the previous
16800                              <1>     ; -current- logical drive's current directory again.)
16801                              <1>
16802 000113FF 89DE                <1>     mov     esi, ebx
16803                              <1>     ; file name is not forced, change directory as temporary
16804 00011401 31C0                <1>     xor     eax, eax
16805                              <1>     ;mov    [FFF_Valid], ah ; 0 ; reset
16806                              <1>     ;call   set_working_path
16807 00011403 E8A0080000          <1>     call    set_working_path_xx
16808 00011408 731D                <1>     jnc     short syschdir_ok
16809 0001140A 21C0                <1>     and     eax, eax  ; 0 -> Bad Path!
16810 0001140C 7505                <1>     jnz     short syschdir_err
16811                              <1>     ; eax = 0
16812                              <1> syschdir_not_found:
16813 0001140E B80C000000          <1>     mov     eax, ERR_DIR_NOT_FOUND ; Directory not found !
16814                              <1> syschdir_err:
16815 00011413 A3[348E0100]        <1>     mov     [u.r0], eax
16816 00011418 A3[A08E0100]        <1>     mov     [u.error], eax
16817 0001141D E857090000          <1>     call    reset_working_path
```

```
16818 00011422 E937B6FFFF          <1>     jmp     error
16819                              <1> syschdir_ok:
16820 00011427 31C0                <1>     xor     eax, eax ; 0
16821 00011429 A3[348E0100]        <1>     mov     [u.r0], eax
16822                              <1>     ;mov    [u.error], eax
16823 0001142E E94BB6FFFF          <1>     jmp     sysret
16824                              <1>
16825                              <1>
16826                              <1> syschmod: ; Get & Change File (or Directory) Attributes
16827                              <1>     ; 26/09/2024 - TRDOS 386 v2.0.9
16828                              <1>     ; 23/07/2022 - TRDOS 386 v2.0.5
16829                              <1>     ; 19/01/2021
16830                              <1>     ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
16831                              <1>     ;
16832                              <1>        ; INPUT ->
16833                              <1>        ;          EBX = File/Directory (ASCIIZ) name address
16834                              <1>     ;          CL = New attributes (if CL < 40h)
16835                              <1>     ;          CL >= 40h -> Get File Attributes
16836                              <1>     ; OUTPUT ->
16837                              <1>     ;          cf = 0 -> EAX = File attributes (in AL)
16838                              <1>     ;          cf = 1 -> Error code in AL
16839                              <1>     ;
16840                              <1>     ; Modified Registers: EAX (at the return of system call)
16841                              <1>     ;
16842                              <1>     ; MSDOS File Attributes:    (bit value of attrib byte)
16843                              <1>     ;      ATTR_READ_ONLY =      01h  (bit 0, 'R')
16844                              <1>     ;      ATTR_HIDDEN    =      02h  (bit 1, 'H')
16845                              <1>     ;      ATTR_SYSTEM    =      04h  (bit 2, 'S')
16846                              <1>     ;      ATTR_VOLUME_ID =      08h  (bit 3)
16847                              <1>     ;      ATTR_DIRECTORY =      10h  (bit 4)
16848                              <1>     ;      ATTR_ARCHIVE   =      20h  (bit 5, 'A')
16849                              <1>     ;      ATTR_LONG_NAME =      ATTR_READONLY |
16850                              <1>     ;                            ATTR_HIDDEN |
16851                              <1>     ;                            ATTR_SYSTEM |
16852                              <1>     ;                            ATTR_VOLUME_ID
16853                              <1>     ;      The upper two bits of attributes must be 0.
16854                              <1>
16855                              <1>     ; Note: * If ATTR_DIRECTORY is set, only directory names
16856                              <1>     ;         will be searched (and S,H,R,A attributeds of
16857                              <1>     ;         the directory will be changed.)
16858                              <1>     ;       * If ATTR_VOLUME_ID is set, 'syschmod' system call
16859                              <1>     ;         will return with 'permission denied' error.
16860                              <1>     ;       * If ATTR_DIRECTORY is not set, only file names
16861                              <1>     ;         will be searched (and S,H,R,A attributes of the
16862                              <1>     ;         file will be changed.)
16863                              <1>     ;
16864                              <1>     ; (Ony Super User can change S,H,R attributes.)
16865                              <1>
16866 00011433 80F940              <1>     cmp     cl, 40h
16867 00011436 7327                <1>     jnb     short syschmod_0
16868                              <1>
16869 00011438 F6C108              <1>     test    cl, 08h ; ATTR_VOLUME_ID
16870 0001143B 750E                <1>     jnz     short syschmod_perm_err
16871                              <1>
16872                              <1>     ; 19/01/2021
16873 0001143D 803D[868E0100]00    <1>     cmp     byte [u.uid], 0  ; root (super user) ?
16874 00011444 7619                <1>     jna     short syschmod_0
16875                              <1>
16876                              <1>     ; Not super user..
16877 00011446 F6C107              <1>     test    cl, 07h ; S,H,R attributes
16878 00011449 7414                <1>     jz      short syschmod_0
16879                              <1>
16880                              <1> syschmod_perm_err:
16881                              <1>     ;mov    dword [u.r0], ERR_PERM_DENIED
16882 0001144B B80B000000          <1>     mov     eax, ERR_PERM_DENIED ; 'permission denied !'
16883 00011450 A3[348E0100]        <1>     mov     [u.r0], eax
16884 00011455 A3[A08E0100]        <1>     mov     [u.error], eax
16885 0001145A E9FFB5FFFF          <1>     jmp     error
16886                              <1>
16887                              <1> syschmod_0:
16888 0001145F 880D[FC800100]      <1>     mov     [Attributes], cl
16889 00011465 89DE                <1>     mov     esi, ebx
16890                              <1>     ; file name is forced, change directory as temporary
16891                              <1>     ;mov    ax, 1
16892                              <1>     ;mov    [FFF_Valid], ah ; 0 ; reset
16893                              <1>     ;call   set_working_path
16894 00011467 E838080000          <1>     call    set_working_path_x
16895 0001146C 7308                <1>     jnc     short syschmod_1
16896 0001146E 21C0                <1>     and     eax, eax  ; 0 -> Bad Path!
16897 00011470 751E                <1>     jnz     short syschmod_err
16898                              <1>     ; eax = 0
16899                              <1> syschmod_path_not_found:
16900                              <1>     ;mov    eax, ERR_INV_PATH_NAME ; 'Bad path name !'
16901                              <1>     ; 23/07/2022
16902 00011472 B013                <1>     mov     al, ERR_INV_PATH_NAME
16903                              <1> ;syschmod_err:
16904                              <1>     ;mov    [u.r0], eax
16905                              <1>     ;mov    [u.error], eax
16906                              <1>     ;call   reset_working_path
16907                              <1>     ;jmp    error
16908                              <1>     ; 23/07/2022
16909 00011474 EB1A                <1>     jmp     short syschmod_err
16910                              <1> syschmod_1:
16911 00011476 B008                <1>     mov     al, 08h ; Except volume labels (& long names)
16912 00011478 A0[FC800100]        <1>     mov     al, [Attributes]
16913 0001147D 2410                <1>     and     al, 10h ;
16914                              <1>     ;mov    esi, FindFile_Name
16915                              <1>     ;mov    ax, 1800h ; Only files
16916                              <1>     ;mov    ax, 0810h ; Only directories
16917 0001147F E8E674FFFF          <1>     call    find_first_file
16918                              <1>     ;jnc    short syschmod_2
16919 00011484 720A                <1>     jc      short syschmod_err
16920                              <1>
16921                              <1>     ;; eax = 2 (File not found !)
16922                              <1>     ;cmp    al, 2 ; ERR_NOT_FOUND
16923                              <1>     ;jne    short syschmod_err
16924                              <1>
16925                              <1>     ;and    byte [Attributes], 10h
16926                              <1>     ;jz     short syschmod_err
16927                              <1>
16928                              <1>     ;; Directory not found !
16929                              <1>     ;mov    al, 3 ; ERR_PATH_NOT_FOUND
16930                              <1>     ;jmp    short syschmod_err
16931                              <1>
16932                              <1> syschmod_2:
16933 00011486 6621D2              <1>     and     dx, dx ; Ambiguous filename chars used sign (DX>0)
16934 00011489 7419                <1>     jz      short syschmod_3
16935 0001148B B81A000000          <1>     mov     eax, ERR_INV_FILE_NAME ; 'invalid file name !'
16936                              <1>        ;jmp       short syschmod_err
16937                              <1> syschmod_err:
16938 00011490 A3[348E0100]        <1>     mov     [u.r0], eax
16939 00011495 A3[A08E0100]        <1>     mov     [u.error], eax
16940 0001149A E8DA080000          <1>     call    reset_working_path
16941 0001149F E9BAB5FFFF          <1>     jmp     error
```

```
16942                                   <1> syschmod_3:
16943                                   <1>     ; EDI = Directory buffer entry offset/address
16944                                   <1>     ; BL = File (or Directory) Attributes
16945                                   <1>     ; mov  bl, [EDI+0Bh]
16946                                   <1>
16947                                   <1>     ; check directory attributes
16948 000114A4 8A3D[FC800100]           <1>     mov    bh, [Attributes] ; new attributes
16949 000114AA 80FF40                   <1>     cmp    bh, 40h  ;>=40 -> get file/directory attributes
16950 000114AD 7325                     <1>     jnb    short syschmod_6
16951                                   <1>
16952                                   <1>     ; set file/directory attributes
16953 000114AF F6C307                   <1>     test   bl, 7 ; system, hidden, readonly
16954 000114B2 7409                     <1>        jz  short syschmod_4
16955                                   <1>
16956                                   <1>     ; 19/01/2021
16957 000114B4 803D[868E0100]00         <1>     cmp    byte [u.uid], 0  ; root (super user) ?
16958 000114BB 778E                     <1>     ja     short syschmod_perm_err
16959                                   <1> syschmod_4:
16960                                   <1> ; 19/12/2025
16961                                   <1> %if 0
16962                                   <1>     cmp    word [edi+DirEntry_NTRes], 01A1h ; Singlix FS
16963                                   <1>     je     short syschmod_7
16964                                   <1> %endif
16965 000114BD 887F0B                  <1>     mov    [edi+0Bh], bh    ; Attributes (New!)
16966                                   <1>
16967 000114C0 C605[697E0100]02         <1>     mov    byte [DirBuff_ValidData], 2 ; modified sign
16968                                   <1>                                        ; to force write
16969 000114C7 E8B598FFFF               <1>     call   save_directory_buffer
16970 000114CC 72C2                     <1>     jc     short syschmod_err
16971                                   <1>
16972                                   <1> syschmod_5:
16973 000114CE 8A1D[FC800100]           <1>     mov    bl, [Attributes]
16974                                   <1> syschmod_6:
16975 000114D4 0FB6C3                   <1>     movzx  eax, bl
16976 000114D7 A3[348E0100]             <1>     mov    [u.r0], eax
16977                                   <1>     ;mov   dword [u.error], 0
16978 000114DC E99DB5FFFF               <1>     jmp    sysret
16979                                   <1>
16980                                   <1> ; 19/12/2025
16981                                   <1> %if 0
16982                                   <1> syschmod_7:
16983                                   <1>     sub    eax, eax
16984                                   <1>        mov     ah, [DirBuff_DRV]
16985                                   <1>     ; 26/09/2024 (BugFix)
16986                                   <1>     sub    ah, 'A'
16987                                   <1>     mov    esi, Logical_DOSDisks
16988                                   <1>        add     esi, eax
16989                                   <1>        cmp     byte [esi+LD_FSType], 0A1h
16990                                   <1>     jnc    short syschmod_8
16991                                   <1>     mov    al, ERR_INV_DATA ; 29 = Invalid Data
16992                                   <1>     jmp    short syschmod_err
16993                                   <1>
16994                                   <1> syschmod_8:
16995                                   <1>     ; BH = New MS-DOS File Attributes
16996                                   <1>     mov    al, bh ; File/Directory Attributes
16997                                   <1>     xor    ah, ah ; Attributes in MS-DOS format sign
16998                                   <1>     call   change_fs_file_attributes
16999                                   <1>     ;jc    syschmod_err
17000                                   <1>     ;jmp   short syschmod_5
17001                                   <1>     ; 23/07/2022
17002                                   <1>     jnc    short syschmod_5
17003                                   <1>     jmp    syschmod_err ; 26/09/2024
17004                                   <1> %endif
17005                                   <1>
17006                                   <1> sysdrive: ; Get/Set Current (Working) Drive (for user)
17007                                   <1>     ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
17008                                   <1>     ;
17009                                   <1>        ; INPUT ->
17010                                   <1>        ;          BL = Logical DOS Drive number (0=A: ... 2=C:)
17011                                   <1>     ;         If BL = 0FFh -> Get Current Drive
17012                                   <1>     ; OUTPUT ->
17013                                   <1>     ;          cf = 0 ->
17014                                   <1>     ;                 AL = Current Drive number
17015                                   <1>     ;                 AH = The Last Logical DOS Drive no.
17016                                   <1>     ;          cf = 1 -> Error code in AL
17017                                   <1>     ;
17018                                   <1>     ; Modified Registers: EAX (at the return of system call)
17019                                   <1>     ;
17020                                   <1>     ; NOTE: If the requested logical dos drive is ready,
17021                                   <1>     ;       it's current current directory will be the user's
17022                                   <1>     ;       (program's) current directory.
17023                                   <1>     ;       (When the program is terminated, MainProg -internal
17024                                   <1>     ;       shell- will reset the previous -current- logical drive
17025                                   <1>     ;        as current drive again).
17026                                   <1>
17027 000114E1 80FBFF                  <1>     cmp    bl, 0FFh
17028 000114E4 7435                     <1>     je     short sysdrive_ok
17029 000114E6 3A1D[5C2E0100]           <1>     cmp    bl, [Last_DOS_DiskNo]
17030 000114EC 771E                     <1>     ja     short sysdrive_err
17031                                   <1>
17032                                   <1>     ; Save current drive and reset mode
17033                                   <1>     ; for 'reset_working_path' procedure (for MainProg)
17034 000114EE 30C0                     <1>     xor    al, al
17035 000114F0 66A3[3C830100]           <1>     mov    [SWP_Mode], ax ; ah = 0
17036 000114F6 A0[42770100]             <1>     mov    al, [Current_Drv]
17037 000114FB FEC4                     <1>     inc    ah ; mov ah, 1
17038 000114FD 66A3[3E830100]           <1>     mov    [SWP_DRV], ax
17039                                   <1>
17040 00011503 88DA                     <1>     mov    dl, bl
17041 00011505 E8D661FFFF               <1>     call   change_current_drive
17042 0001150A 730F                     <1>     jnc    short sysdrive_ok
17043                                   <1> sysdrive_err:
17044 0001150C C705[348E0100]0F00-      <1>     mov    dword [u.r0], ERR_DRV_NOT_RDY ; 'drive not ready !'
17044 00011514 0000                     <1>
17045 00011516 E943B5FFFF               <1>     jmp    error
17046                                   <1> sysdrive_ok:
17047 0001151B A0[42770100]             <1>     mov    al, [Current_Drv]
17048 00011520 8A25[5C2E0100]           <1>     mov    ah, [Last_DOS_DiskNo]
17049 00011526 A3[348E0100]             <1>     mov    [u.r0], eax
17050 0001152B E94EB5FFFF               <1>     jmp    sysret
17051                                   <1>
17052                                   <1> sysdir: ; Get Current (Working) Drive & Directory (for user)
17053                                   <1>     ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
17054                                   <1>     ;
17055                                   <1>        ; INPUT ->
17056                                   <1>        ;          EBX = Current directory name buffer address
17057                                   <1>     ;                (Buffer length = 92 bytes)
17058                                   <1>     ; OUTPUT ->
17059                                   <1>     ;          AL = Current drive (0=A: .. 2=C:)
17060                                   <1>     ;         If CF = 1 -> AL = error code
17061                                   <1>     ;
17062                                   <1>     ; Modified Registers: EAX (at the return of system call)
17063                                   <1>     ;
17064                                   <1>     ; Note: Required directory name buffer length may be
```

```
17065                                  <1>     ;       <= 92 bytes for current TRDOS 386 version.
17066                                  <1>     ;       (7*12 name chars + 7 slash + 0)
17067                                  <1>
17068 00011530 89E5                    <1>     mov     ebp, esp
17069 00011532 83EC60                  <1>     sub     esp, 96
17070 00011535 53                      <1>     push    ebx ; User's buffer address
17071 00011536 30D2                    <1>     xor     dl, dl ; 0 = current drive
17072 00011538 E8448EFFFF              <1>     call    get_current_directory
17073 0001153D 72CD                    <1>     jc      short sysdrive_err ; 'drive not ready !' error
17074 0001153F 89E6                    <1>     mov     esi, esp ; System's buffer address
17075 00011541 5F                      <1>     pop     edi  ; User's buffer address
17076                                  <1>     ; ecx = transfer (byte) count (<=92)
17077 00011542 E8A3F5FFFF              <1>     call    transfer_to_user_buffer
17078 00011547 89EC                    <1>     mov     esp, ebp
17079 00011549 730F                    <1>     jnc     short sysdir_ok
17080                                  <1> sysdir_err:
17081 0001154B C705[348E0100]2E00-     <1>     mov     dword [u.r0], ERR_BUFFER  ; 'buffer error !'
17081 00011553 0000                    <1>
17082 00011555 E904B5FFFF              <1>     jmp     error
17083                                  <1> sysdir_ok:
17084 0001155A 8A0D[42770100]          <1>     mov     cl, [Current_Drv]
17085 00011560 890D[348E0100]          <1>     mov     [u.r0], ecx
17086 00011566 E913B5FFFF              <1>     jmp     sysret
17087                                  <1>
17088                                  <1> sysldrvt: ; Get copy of Logical DOS Drive Description Table
17089                                  <1>     ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
17090                                  <1>     ;
17091                                  <1>        ; INPUT ->
17092                                  <1>     ;          BL = Logical DOS drive number (zero based)
17093                                  <1>        ;          ECX = Logical DOS drv desc table buffer addr
17094                                  <1>     ;              (Buffer length = 256 bytes)
17095                                  <1>     ; OUTPUT ->
17096                                  <1>     ;          cf = 0 ->
17097                                  <1>     ;                 AL = Current Drive number
17098                                  <1>     ;                 AH = The Last Logical DOS Drive no.
17099                                  <1>     ;          cf = 1 -> Error code in AL
17100                                  <1>     ;                 AH = The Last Logical DOS Drive no.
17101                                  <1>     ;
17102                                  <1>     ; Modified Registers: EAX (at the return of system call)
17103                                  <1>     ;
17104                                  <1>     ; Note: Required description table buffer length is
17105                                  <1>     ;       256 bytes for current TRDOS 386 version.
17106                                  <1>
17107 0001156B 89CF                    <1>     mov     edi, ecx ; Destination address (user space)
17108 0001156D 88DC                    <1>     mov     ah, bl
17109 0001156F 30C0                    <1>     xor     al, al
17110 00011571 BE00010900              <1>     mov     esi, Logical_DOSDisks
17111 00011576 01C6                    <1>     add     esi, eax ; Source address (system space)
17112 00011578 B900010000              <1>     mov     ecx, 256 ; Byte count
17113                                  <1>                      ; Logical Dos Drv Desc Table size
17114 0001157D E868F5FFFF              <1>     call    transfer_to_user_buffer
17115 00011582 72C7                    <1>     jc      short sysdir_err
17116 00011584 8A2D[5C2E0100]          <1>     mov     ch, [Last_DOS_DiskNo]
17117 0001158A EBCE                    <1>     jmp     short sysdir_ok
17118                                  <1>
17119                                  <1> systime: ; Get System Date&Time
17120                                  <1>     ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
17121                                  <1>     ;
17122                                  <1>        ; INPUT -> BL =
17123                                  <1>     ;          0 = Get Date&Time in Unix/Epoch format
17124                                  <1>     ;          1 = Get Time in MSDOS format
17125                                  <1>     ;          2 = Get Date in MSDOS format
17126                                  <1>     ;          3 = Get Date&Time in MSDOS format
17127                                  <1>     ;          4 & other values =
17128                                  <1>     ;             System timer ticks will be returned
17129                                  <1>     ;             in EAX and Carry Flag will be set.
17130                                  <1>     ;             (CF will not be set if BL = 4)
17131                                  <1>     ; OUTPUT ->
17132                                  <1>     ;       For BL input = 3
17133                                  <1>     ;          EAX = Current Time (RTC)
17134                                  <1>     ;             AL = Second (DL in MSDOS)
17135                                  <1>     ;             AH = Minute (CL in MSDOS)
17136                                  <1>     ;             HW of EAX = Hour (CH in MSDOS)
17137                                  <1>     ;          EDX = Current System Date (RTC)
17138                                  <1>     ;             DL = Day (DL in MSDOS)
17139                                  <1>     ;             DH = Month (DH in MSDOS)
17140                                  <1>     ;             HW of EDX = Year (CX in MSDOS)
17141                                  <1>     ;
17142                                  <1>     ;       For BL input = 2
17143                                  <1>     ;          EAX = Current System Date (RTC)
17144                                  <1>     ;             DL = Day (DL in MSDOS)
17145                                  <1>     ;             DH = Month (DH in MSDOS)
17146                                  <1>     ;             HW of EDX = Year (CX in MSDOS)
17147                                  <1>     ;
17148                                  <1>     ;       For BL input = 1
17149                                  <1>     ;          EAX = Current Time (RTC)
17150                                  <1>     ;             AL = Second (DL in MSDOS)
17151                                  <1>     ;             AH = Minute (CL in MSDOS)
17152                                  <1>     ;             HW of EAX = Hour (CH in MSDOS)
17153                                  <1>     ;
17154                                  <1>     ;       For BL input = 0
17155                                  <1>     ;          EAX = Unix (Epoch) Time Ticks/Seconds
17156                                  <1>     ;
17157                                  <1>     ;       For BL input  = 4
17158                                  <1>     ;          EAX = System timer ticks
17159                                  <1>     ;
17160                                  <1>     ;       If CF = 1 (for other values of BL input)
17161                                  <1>     ;          EAX = System timer ticks (no error code!)
17162                                  <1>     ;
17163                                  <1>     ; Modified Registers: EAX, (EDX)
17164                                  <1>     ;                (at the return of system call)
17165                                  <1>     ;
17166                                  <1>
17167 0001158C 20DB                    <1>     and     bl, bl
17168 0001158E 750F                    <1>     jnz     short systime_1
17169 00011590 E83C59FFFF              <1>     call    epoch
17170                                  <1> systime_0:
17171 00011595 A3[348E0100]            <1>     mov     [u.r0], eax
17172 0001159A E9DFB4FFFF              <1>     jmp     sysret
17173                                  <1> systime_1:
17174 0001159F 80FB04                  <1>     cmp     bl, 4
17175 000115A2 7211                    <1>     jb      short systime_2
17176 000115A4 A1[00770100]            <1>     mov     eax, [TIMER_LH] ; 18.2 Hz timer ticks
17177                                  <1>                             ; Note: [TIMER_LH] may be set
17178                                  <1>                             ; to wrong timer value due to
17179                                  <1>                             ; program functions.
17180                                  <1>                             ; (This value must not be
17181                                  <1>                             ; accepted as [TIMER_LH]/18.2
17182                                  <1>                             ; seconds since the midnight.)
17183 000115A9 76EA                    <1>     jna     short systime_0
17184 000115AB A3[348E0100]            <1>     mov     [u.r0], eax
17185 000115B0 E9A9B4FFFF              <1>     jmp     error ; cf = 1 & [u.r0] = eax = timer ticks
17186                                  <1>
17187                                  <1> systime_2:
```

```
17188                                   <1>     ;push   ebx
17189 000115B5 E89158FFFF               <1>     call    get_rtc_date_time
17190                                   <1>     ;pop    ebx
17191 000115BA F6C301                   <1>     test    bl, 1
17192 000115BD 7429                     <1>     jz      short systime_4
17193 000115BF 30E4                     <1>     xor     ah, ah
17194 000115C1 A0[4A730100]             <1>     mov     al, [hour]
17195 000115C6 88C2                     <1>     mov     dl, al
17196 000115C8 C1E010                   <1>     shl     eax, 16
17197 000115CB A0[4E730100]             <1>     mov     al, [second]
17198 000115D0 8A25[4C730100]           <1>     mov     ah, [minute]
17199 000115D6 F6C302                   <1>     test    bl, 2
17200 000115D9 74BA                     <1>     jz      short systime_0
17201                                   <1>     ; Check time & date match risk
17202                                   <1>     ; (23:59:59 may cause to wrong
17203                                   <1>     ; date -new day with previous date-...)
17204 000115DB 80FA17                   <1>     cmp     dl, 23
17205 000115DE 7206                     <1>     jb      short systime_3
17206 000115E0 663D3B3B                 <1>     cmp     ax, (59*256)+59 ; if hour is 23:59:59
17207 000115E4 73CF                     <1>     jnb     short systime_2 ; wait for 1 second
17208                                   <1> systime_3:
17209                                   <1>     ; eax = time
17210 000115E6 89C6                     <1>     mov     esi, eax
17211                                   <1> systime_4:
17212 000115E8 66A1[44730100]           <1>     mov     ax, [year]
17213 000115EE C1E010                   <1>     shl     eax, 16
17214 000115F1 A0[48730100]             <1>     mov     al, [day]
17215 000115F6 8A25[46730100]           <1>     mov     ah, [month]
17216                                   <1>     ; eax = date
17217 000115FC 80E301                   <1>     and     bl, 1
17218 000115FF 7494                     <1>     jz      short systime_0
17219 00011601 96                       <1>     xchg    esi, eax
17220                                   <1>     ; eax = time, esi = date
17221 00011602 8B2D[308E0100]           <1>     mov     ebp, [u.usp]  ; EBP points to user's registers
17222                                   <1>     ; (user) edx <-- (system) esi
17223 00011608 897514                   <1>     mov     [ebp+20], esi ; return to user with EDX value
17224 0001160B EB88                     <1>     jmp     short systime_0
17225                                   <1>
17226                                   <1> sysstime: ; Set System Date&Time
17227                                   <1>     ; 31/12/2017
17228                                   <1>     ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
17229                                   <1>     ;
17230                                   <1>       ; INPUT -> BL =
17231                                   <1>     ;         0 = Set Date&Time in Unix/Epoch format
17232                                   <1>     ;         1 = Set Time in MSDOS format
17233                                   <1>     ;         2 = Set Date in MSDOS format
17234                                   <1>     ;         3 = Set Date&Time in MSDOS format
17235                                   <1>     ;         4 = Set System Timer (Ticks)
17236                                   <1>     ;         5 = Convert/Save current time to/as
17237                                   <1>     ;            18.2 Hz system timer ticks
17238                                   <1>     ;         6 = Convert MSDOS Date&Time to UNIX format
17239                                   <1>     ;            without setting system date&time ; (test)
17240                                   <1>     ;         7 = Convert UNIX Date&Time to MSDOS format
17241                                   <1>     ;            without setting system date&time ; (test)
17242                                   <1>     ;        8-0FFh = invalid !
17243                                   <1>     ;       ECX = Time (or Timer) value in selected format
17244                                   <1>     ;       EDX = Date value in MSDOS format if BL=2,3,6
17245                                   <1>     ;
17246                                   <1>     ; OUTPUT ->
17247                                   <1>     ;      If CF = 0 ->
17248                                   <1>     ;         EAX = Set value
17249                                   <1>     ;      If CF = 1 -> (invalid BL input)
17250                                   <1>     ;         EAX = Ticks count [TIMER_LH]
17251                                   <1>     ;
17252                                   <1>
17253 0001160D 20DB                     <1>     and     bl, bl ; 0
17254 0001160F 7511                     <1>     jnz     short sysstime_0
17255 00011611 89C8                     <1>     mov     eax, ecx
17256 00011613 E83E59FFFF               <1>     call    convert_from_epoch
17257 00011618 E8DE59FFFF               <1>     call    set_rtc_date_time
17258 0001161D E95CB4FFFF               <1>     jmp     sysret
17259                                   <1> sysstime_0:
17260 00011622 80FB08                   <1>     cmp     bl, 8
17261 00011625 722D                     <1>     jb      short sysstime_1
17262                                   <1>     ; invalid input (>7)
17263 00011627 A1[00770100]             <1>     mov     eax, [TIMER_LH] ; 18.2 Hz timer ticks
17264                                   <1>                             ; Note: [TIMER_LH] may be set
17265                                   <1>                             ; to wrong timer value due to
17266                                   <1>                             ; program functions.
17267                                   <1>                             ; (This value must not be
17268                                   <1>                             ; accepted as [TIMER_LH]/18.2
17269                                   <1>                             ; seconds since the midnight.)
17270 0001162C A3[348E0100]             <1>     mov     [u.r0], eax
17271 00011631 E928B4FFFF               <1>     jmp     error ; cf = 1 & [u.r0] = eax = timer ticks
17272                                   <1>
17273                                   <1> sysstime_8:
17274                                   <1>     ; BL = 7
17275 00011636 89C8                     <1>     mov     eax, ecx ; seconds since 1/1/1970 00:00:00
17276 00011638 E81959FFFF               <1>     call    convert_from_epoch
17277 0001163D 30E4                     <1>     xor     ah, ah
17278 0001163F A0[4A730100]             <1>     mov     al, [hour]
17279 00011644 C1E010                   <1>     shl     eax, 16
17280 00011647 A0[4E730100]             <1>     mov     al, [second]
17281 0001164C 8A25[4C730100]           <1>     mov     ah, [minute]
17282 00011652 EB92                     <1>     jmp     short systime_3
17283                                   <1>
17284                                   <1> sysstime_1:
17285 00011654 80FB04                   <1>     cmp     bl, 4
17286 00011657 743F                     <1>     je      short sysstime_2 ; set system timer ticks
17287 00011659 80FB05                   <1>     cmp     bl, 5
17288 0001165C 754B                     <1>     jne     short sysstime_4
17289                                   <1>     ; convert current time to system timer ticks (18.2Hz)
17290 0001165E E8E857FFFF               <1>     call    get_rtc_date_time
17291 00011663 0FB60D[4A730100]         <1>     movzx   ecx, byte [hour]
17292 0001166A B8100E0000               <1>     mov     eax, 60*60 ; 1 hour = 3600 seconds
17293 0001166F F7E1                     <1>     mul     ecx
17294 00011671 89C3                     <1>     mov     ebx, eax
17295 00011673 B13C                     <1>     mov     cl, 60  ; 1 minute = 60 seconds
17296 00011675 0FB605[4C730100]         <1>     movzx   eax, byte [minute]
17297 0001167C F7E1                     <1>     mul     ecx
17298 0001167E 01D8                     <1>     add     eax, ebx
17299 00011680 8A0D[4E730100]           <1>     mov     cl, [second]
17300 00011686 01C8                     <1>     add     eax, ecx
17301 00011688 B1B6                     <1>     mov     cl, 182
17302 0001168A F7E1                     <1>     mul     ecx
17303 0001168C 83C009                   <1>     add     eax, 9
17304 0001168F 83D200                   <1>     adc     edx, 0
17305 00011692 B10A                     <1>     mov     cl, 10
17306 00011694 F7F1                     <1>     div     ecx
17307                                   <1>     ; eax = ((182*seconds)+9)/10
17308 00011696 89C1                     <1>     mov     ecx, eax
17309                                   <1> sysstime_2:
17310 00011698 890D[00770100]           <1>     mov     [TIMER_LH], ecx ; 18.2 * seconds
17311                                   <1> sysstime_3:
```

```
17312 0001169E 890D[348E0100]      <1>     mov     [u.r0], ecx
17313 000116A4 E9D5B3FFFF          <1>     jmp     sysret
17314                              <1> sysstime_4:
17315 000116A9 80FB06              <1>     cmp     bl, 6
17316 000116AC 7788                <1>     ja      short sysstime_8
17317                              <1>
17318 000116AE 890D[348E0100]      <1>     mov     [u.r0], ecx
17319                              <1>
17320 000116B4 880D[4E730100]      <1>     mov     [second], cl
17321 000116BA 882D[4C730100]      <1>     mov     [minute], ch
17322 000116C0 C1E910              <1>     shr     ecx, 16
17323 000116C3 880D[4A730100]      <1>     mov     [hour], cl
17324                              <1>     ; BL = 1,2,3,6
17325 000116C9 80FB01              <1>     cmp     bl, 1
17326 000116CC 762A                <1>     jna     short sysstime_5
17327                              <1>     ; BL = 2,3,6
17328 000116CE 8815[48730100]      <1>     mov     [day], dl
17329 000116D4 8835[46730100]      <1>     mov     [month], dh
17330 000116DA C1EA10              <1>     shr     edx, 16
17331 000116DD 668915[44730100]    <1>     mov     [year], dx
17332 000116E4 80E303              <1>     and     bl, 3
17333 000116E7 742D                <1>     jz      short sysstime_7 ; 6
17334                              <1>     ; BL = 2,3
17335 000116E9 F6C301              <1>     test    bl, 1
17336 000116EC 7419                <1>     jz      short sysstime_6 ; 2
17337                              <1>     ; BL = 3
17338 000116EE E80859FFFF          <1>     call    set_rtc_date_time
17339 000116F3 E986B3FFFF          <1>     jmp     sysret
17340                              <1> sysstime_5:
17341                              <1>     ; BL = 1
17342 000116F8 E83F59FFFF          <1>     call    set_time_bcd
17343 000116FD E87D4CFFFF          <1>     call    set_rtc_time
17344 00011702 E977B3FFFF          <1>     jmp     sysret
17345                              <1> sysstime_6:
17346                              <1>     ; BL = 2
17347 00011707 E80359FFFF          <1>     call    set_date_bcd
17348 0001170C E8D14CFFFF          <1>     call    set_rtc_date
17349 00011711 E968B3FFFF          <1>     jmp     sysret
17350                              <1> sysstime_7:
17351                              <1>     ; BL = 6
17352                              <1>     ; [year], [month], [day],
17353                              <1>     ; [hour], [minute], [second]
17354 00011716 E8BB57FFFF          <1>     call    convert_to_epoch
17355 0001171B 89C1                <1>     mov     ecx, eax ; seconds since 1/1/1970 00:00:00
17356 0001171D E97CFFFFFF          <1>     jmp     sysstime_3
17357                              <1>
17358                              <1> sysrename: ; Rename File (or Directory)
17359                              <1>     ; 19/12/2025 - TRDOS 386 v2.0.10
17360                              <1>     ; 08/08/2022
17361                              <1>     ; 23/07/2022 - TRDOS 386 v2.0.5
17362                              <1>     ; 19/01/2021
17363                              <1>     ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
17364                              <1>     ;
17365                              <1>         ; INPUT ->
17366                              <1>         ;          EBX = File/Directory (ASCIIZ) name address
17367                              <1>     ;         ECX = New name (in same dir, no path name)
17368                              <1>     ; OUTPUT ->
17369                              <1>     ;          cf = 0 -> EAX = 0
17370                              <1>     ;          cf = 1 -> Error code in AL
17371                              <1>
17372                              <1>     ; 19/01/2021
17373 00011722 803D[868E0100]00    <1>     cmp     byte [u.uid], 0  ; root (super user) ?
17374                              <1>     ;jna    short sysrename_0
17375                              <1>     ; 23/07/2022
17376 00011729 7717                <1>     ja      short sysrename_perm_err
17377                              <1>
17378                              <1> ;sysrename_perm_err:
17379                              <1> ;   ;mov    dword [u.r0], ERR_PERM_DENIED
17380                              <1> ;   mov     eax, ERR_PERM_DENIED ; 'permission denied !'
17381                              <1> ;   mov     [u.r0], eax
17382                              <1> ;   mov     [u.error], eax
17383                              <1> ;   jmp     error
17384                              <1>
17385                              <1> sysrename_0:
17386 0001172B 51                  <1>     push    ecx ; new file name address (in user space)
17387 0001172C 89DE                <1>     mov     esi, ebx
17388                              <1>     ; file name is forced, change directory as temporary
17389                              <1>     ;mov    ax, 1
17390                              <1>     ;mov    [FFF_Valid], ah ; 0 ; reset
17391                              <1>     ;call   set_working_path
17392 0001172E E871050000          <1>     call    set_working_path_x
17393 00011733 7321                <1>     jnc     short sysrename_1
17394 00011735 21C0                <1>     and     eax, eax  ; 0 -> Bad Path!
17395 00011737 753B                <1>     jnz     short sysrename_err
17396                              <1>     ; eax = 0
17397                              <1> sysrename_path_not_found:
17398 00011739 B813000000          <1>     mov     eax, ERR_INV_PATH_NAME ; 'Bad path name !'
17399                              <1>     ; 23/07/2022
17400 0001173E B013                <1>     mov     al, ERR_INV_PATH_NAME
17401 00011740 EB32                <1>     jmp     short sysrename_err
17402                              <1> ;sysrename_err:
17403                              <1> ;   pop     ecx ; new file name address (in user space)
17404                              <1> ;sysrename_error:
17405                              <1> ;   mov     [u.r0], eax
17406                              <1> ;   mov     [u.error], eax
17407                              <1> ;   call    reset_working_path
17408                              <1> ;   jmp     error
17409                              <1>
17410                              <1>     ; 23/07/2022
17411                              <1> sysrename_perm_err:
17412                              <1>     ;mov    dword [u.r0], ERR_PERM_DENIED
17413 00011742 B80B000000          <1>     mov     eax, ERR_PERM_DENIED ; 'permission denied !'
17414 00011747 A3[348E0100]        <1>     mov     [u.r0], eax
17415 0001174C A3[A08E0100]        <1>     mov     [u.error], eax
17416 00011751 E908B3FFFF          <1>     jmp     error
17417                              <1>
17418                              <1> sysrename_1:
17419 00011756 B008                <1>     mov     al, 08h ; Except volume labels (& long names)
17420 00011758 A0[FC800100]        <1>     mov     al, [Attributes]
17421 0001175D 2410                <1>     and     al, 10h ;
17422                              <1>     ;mov    esi, FindFile_Name
17423                              <1>     ;mov    ax, 1800h ; Only files
17424                              <1>     ;mov    ax, 0810h ; Only directories
17425 0001175F 66B80008            <1>     mov     ax, 0800h ; Find File or Directory
17426 00011763 E80272FFFF          <1>     call    find_first_file
17427                              <1>     ;jnc    short sysrename_2
17428 00011768 720A                <1>     jc      short sysrename_err
17429                              <1> sysrename_2:
17430                              <1>     ; ESI = Directory Entry (FindFile_DirEntry) Location
17431                              <1>     ; EDI = Directory Buffer Directory Entry Location
17432                              <1>     ; EAX = File Size
17433                              <1>     ;  BL = Attributes of The File/Directory
17434                              <1>     ;  BH = Long Name Yes/No Status (>0 is YES)
17435                              <1>     ;  DX > 0 : Ambiguous filename chars are used
```

```
17436                                   <1>
17437 0001176A 6621D2                  <1>     and     dx, dx ; Ambiguous filename chars used sign (DX>0)
17438 0001176D 741A                    <1>     jz      short sysrename_3
17439 0001176F B81A000000              <1>     mov     eax, ERR_INV_FILE_NAME ; 'invalid file name !'
17440                                   <1>     ;jmp    short sysrename_err
17441                                   <1>     ; 23/07/2022
17442                                   <1> sysrename_err:
17443 00011774 59                      <1>     pop     ecx ; new file name address (in user space)
17444                                   <1> sysrename_error:
17445 00011775 A3[348E0100]            <1>     mov     [u.r0], eax
17446 0001177A A3[A08E0100]            <1>     mov     [u.error], eax
17447 0001177F E8F5050000              <1>     call    reset_working_path
17448 00011784 E9D5B2FFFF              <1>     jmp     error
17449                                   <1> sysrename_3:
17450                                   <1>     ; EDI = Directory buffer entry offset/address
17451                                   <1>     ; BL = File (or Directory) Attributes
17452                                   <1>     ; mov  bl, [EDI+0Bh]
17453                                   <1>
17454 00011789 5A                      <1>     pop     edx ; new file name address (in user space)
17455                                   <1>
17456                                   <1>     ; check file/directory attributes
17457 0001178A F6C307                  <1>     test    bl, 7 ; system, hidden, readonly
17458 0001178D 75B3                    <1>         jnz short sysrename_perm_err
17459                                   <1> sysrename_4:
17460                                   <1>     ; 19/12/2025
17461                                   <1>     ;cmp    word [edi+DirEntry_NTRes], 01A1h ; Singlix FS
17462                                   <1>     ;je     short sysrename_perm_err ; -temporary!-
17463                                   <1>
17464                                   <1>     ; save old file name & file info (FFF structure)
17465 0001178F BE[E27F0100]            <1>     mov     esi, FindFile_Drv
17466 00011794 BF[2C810100]            <1>     mov     edi, SourceFile_Drv
17467                                   <1>     ;mov    ecx, 128/4
17468                                   <1>     ; 23/07/2022
17469 00011799 29C9                    <1>     sub     ecx, ecx
17470 0001179B B120                    <1>     mov     cl, 128/4
17471 0001179D F3A5                    <1>     rep     movsd
17472                                   <1>
17473 0001179F 89D6                    <1>     mov     esi, edx ; new file name address (in user space)
17474 000117A1 BF[AC810100]            <1>     mov     edi, DestinationFile_Drv
17475 000117A6 E8C391FFFF              <1>     call    parse_path_name
17476 000117AB 72C8                    <1>     jc      short sysrename_error ; eax = 1 (Bad file name)
17477                                   <1>
17478                                   <1>     ; same drive ?
17479 000117AD A0[E27F0100]            <1>     mov     al, [FindFile_Drv]
17480 000117B2 3A05[AC810100]          <1>     cmp     al, [DestinationFile_Drv]
17481                                   <1>     ;jne    short sysrename_perm_err ; Permission denied
17482 000117B8 7509                    <1>     jne     short sysrename_5 ; Bad file name
17483                                   <1>
17484                                   <1>     ; no path name !? (rename file in same directory)
17485 000117BA 803D[AD810100]20        <1>     cmp     byte [DestinationFile_Directory], 20h
17486 000117C1 7607                    <1>     jna     short sysrename_6
17487                                   <1> sysrename_5:
17488 000117C3 B801000000              <1>     mov     eax, ERR_BAD_CMD_ARG ; 1 = Bad file name
17489                                   <1>                                  ; (Bad argument)
17490 000117C8 EBAB                    <1>     jmp     short sysrename_error
17491                                   <1> sysrename_6:
17492 000117CA 803D[EE810100]20        <1>     cmp     byte [DestinationFile_Name], 20h
17493 000117D1 76F0                    <1>     jna     short sysrename_5
17494                                   <1>
17495 000117D3 BE[EE810100]            <1>     mov     esi, DestinationFile_Name
17496 000117D8 E8EC74FFFF              <1>     call    check_filename ; is it a valid msdos file name?
17497 000117DD 7296                    <1>     jc      short sysrename_error ; 26 = ERR_INV_FILE_NAME
17498                                   <1>
17499                                   <1>     ;mov    esi, DestinationFile_Name
17500 000117DF 66B80008                <1>     mov     ax, 0800h ; Find File or Directory
17501 000117E3 E88271FFFF              <1>     call    find_first_file
17502 000117E8 7207                    <1>     jc      short sysrename_7
17503                                   <1>
17504 000117EA B80E000000              <1>     mov     eax, ERR_FILE_EXISTS  ; file already exists !
17505                                   <1> jmp_sysrename_err: ; 08/08/2022
17506 000117EF EB84                    <1>     jmp     sysrename_error
17507                                   <1> sysrename_7:
17508                                   <1>     ; eax = 2 (File not found !)
17509 000117F1 3C02                    <1>     cmp     al, 2 ; ERR_NOT_FOUND
17510                                   <1>     ;jne    short sysrename_error
17511                                   <1>     ; 08/08/2022
17512 000117F3 75FA                    <1>     jne     short jmp_sysrename_err
17513                                   <1>
17514                                   <1>     ; 31/12/2017
17515                                   <1>     ; Following code is also part of 'rename_file' in
17516                                   <1>     ; 'trdosk3.s' (MainProg's 'rename' command) ; 13/11/2017
17517 000117F5 BE[EE810100]            <1>     mov     esi, DestinationFile_Name ; (Rename_NewName)
17518 000117FA 668B0D[A6810100]        <1>     mov     cx, [SourceFile_DirEntryNumber]
17519 00011801 66A1[92810100]          <1>     mov     ax, [SourceFile_DirEntry+20] ; First Cluster, HW
17520 00011807 C1E010                  <1>     shl     eax, 16
17521 0001180A 66A1[98810100]          <1>     mov     ax, [SourceFile_DirEntry+26] ; First Cluster, LW
17522 00011810 0FB61D[7B810100]        <1>     movzx   ebx, byte [SourceFile_LongNameEntryLength]
17523 00011817 E80199FFFF              <1>     call    rename_directory_entry
17524                                   <1>     ;jc     short sysrename_error
17525                                   <1>     ; 08/08/2022
17526 0001181C 72D1                    <1>     jc      short jmp_sysrename_err
17527                                   <1>     ;xor    eax, eax
17528 0001181E A3[348E0100]            <1>     mov     [u.r0], eax ; 0
17529                                   <1>     ;mov    [u.error], eax
17530 00011823 E851050000              <1>     call    reset_working_path
17531 00011828 E951B2FFFF              <1>     jmp     sysret
17532                                   <1>
17533                                   <1> sysmem: ; Get Total&Free Memory amount
17534                                   <1>     ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
17535                                   <1>     ;
17536                                   <1>         ; INPUT ->
17537                                   <1>     ;       none
17538                                   <1>     ; OUTPUT ->
17539                                   <1>     ;       EAX = Total memory count (in bytes)
17540                                   <1>     ;       EBX = Virtually available memory amount (in bytes)
17541                                   <1>     ;             = 4GB - CORE (4MB)
17542                                   <1>     ;       ECX = Free memory count (in bytes)
17543                                   <1>     ;       EDX = Calculated free memory count (in bytes)
17544                                   <1>
17545 0001182D A1[84760100]            <1>     mov     eax, [memory_size] ; in pages
17546 00011832 C1E00C                  <1>     shl     eax, 12           ; in bytes
17547 00011835 A3[348E0100]            <1>     mov     [u.r0], eax
17548 0001183A E8F829FFFF              <1>     call    calc_free_mem
17549                                   <1>     ; edx = calculated free pages
17550                                   <1>     ; ecx = 0
17551 0001183F 8B2D[308E0100]          <1>     mov     ebp, [u.usp]  ; EBP points to user's registers
17552 00011845 C745100000C0FF          <1>     mov     dword [ebp+16], ECORE ; EBX (for user)
17553                                   <1>                                   ; 0FFC00000h ; 4GB - 4MB
17554 0001184C C1E20C                  <1>     shl     edx, 12
17555 0001184F 895514                  <1>     mov     [ebp+20], edx ; EDX (for user)
17556 00011852 8B0D[88760100]          <1>     mov     ecx, [free_pages]
17557 00011858 C1E10C                  <1>     shl     ecx, 12 ; free bytes
17558 0001185B 894D18                  <1>     mov     [ebp+24], ecx ; ECX (for user)
17559                                   <1>     ;mov    [free_pages], edx
```

```
17560 0001185E E91BB2FFFF          <1>     jmp     sysret
17561                              <1>
17562                              <1> sysprompt:
17563                              <1>     ; Set TRDOS 386 Command Interpreter (MainProg) prompt
17564                              <1>     ; 30/07/2022 (TRDOS 386 Kernel v2.0.5)
17565                              <1>     ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
17566                              <1>     ;
17567                              <1>        ; INPUT ->
17568                              <1>     ;       EBX = 0 -> use default prompt
17569                              <1>     ;       EBX > 0 -> prompt string (ASCIIZ) address
17570                              <1>     ;                  (Max. 11 characters except ZERO tail)
17571                              <1>     ; OUTPUT ->
17572                              <1>     ;       (EAX = 0)
17573                              <1>     ;       CF = 0 -> Successful
17574                              <1>     ;       CF = 1 -> Failed
17575                              <1>
17576 00011863 21DB               <1>     and     ebx, ebx
17577 00011865 750A               <1>     jnz     short sysprompt_0
17578                              <1>
17579 00011867 E8E56AFFFF         <1>     call    default_command_prompt ; '['+'TRDOS'+']'
17580 0001186C E90DB2FFFF         <1>     jmp     sysret
17581                              <1>
17582                              <1> sysprompt_0:
17583 00011871 31C0               <1>     xor     eax, eax
17584 00011873 A3[348E0100]       <1>     mov     [u.r0], eax
17585 00011878 89DE               <1>     mov     esi, ebx
17586                              <1>     ;mov    ecx, 12
17587                              <1>     ; 30/07/2022
17588 0001187A 31C9               <1>     xor     ecx, ecx
17589 0001187C B10C               <1>     mov     cl, 12
17590 0001187E 89E5               <1>     mov     ebp, esp
17591 00011880 29CC               <1>     sub     esp, ecx
17592 00011882 49                 <1>     dec     ecx ; 11
17593 00011883 89E7               <1>     mov     edi, esp
17594 00011885 E8AAF2FFFF         <1>     call    transfer_from_user_buffer
17595 0001188A 7211               <1>     jc      short sysprompt_err
17596 0001188C 803E20             <1>     cmp     byte [esi], 20h
17597 0001188F 760C               <1>     jna     short sysprompt_err
17598 00011891 E8CD6AFFFF         <1>     call    set_command_prompt
17599 00011896 89EC               <1>     mov     esp, ebp
17600 00011898 E9E1B1FFFF         <1>     jmp     sysret
17601                              <1> sysprompt_err:
17602                              <1> syspath_err:
17603 0001189D 89EC               <1>     mov     esp, ebp
17604 0001189F E9BAB1FFFF         <1>     jmp     error
17605                              <1>
17606                              <1> syspath:
17607                              <1>     ; Get/Set Run Path
17608                              <1>     ;
17609                              <1>     ; 30/07/2022 (TRDOS 386 Kernel v2.0.5)
17610                              <1>     ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
17611                              <1>     ;
17612                              <1>        ; INPUT ->
17613                              <1>     ;       EBX = 0 -> get path (to buffer address in ECX)
17614                              <1>     ;       EBX > 0 -> set path
17615                              <1>     ;           EBX = Path string buffer address (ASCIIZ)
17616                              <1>     ;                 (Path description except 'PATH=')
17617                              <1>     ;       ECX = Buffer address (if EBX = 0)
17618                              <1>     ;             (ECX will not be used if EBX > 0)
17619                              <1>     ;       DL = Buffer size (0 = 256 byte)
17620                              <1>     ;
17621                              <1>     ; OUTPUT ->
17622                              <1>     ;       CF = 0 -> Successful (EAX = String length)
17623                              <1>     ;       CF = 1 -> Failed (EAX = 0)
17624                              <1>     ;
17625                              <1>     ; NOTE: 'PATH=' or 'PATH' must be excluded
17626                              <1>     ;  (It must not be at the beginning of the string.)
17627                              <1>
17628 000118A4 89E5               <1>     mov     ebp, esp
17629 000118A6 81EC00010000       <1>     sub     esp, 256
17630 000118AC 89E7               <1>     mov     edi, esp
17631                              <1>
17632 000118AE 31C0               <1>     xor     eax, eax
17633 000118B0 A3[348E0100]       <1>     mov     [u.r0], eax
17634                              <1>
17635 000118B5 21DB               <1>     and     ebx, ebx
17636 000118B7 752A               <1>     jnz     short syspath_0
17637                              <1>
17638                              <1>     ; EBX = 0 -> get run path
17639 000118B9 89CB               <1>     mov     ebx, ecx ; buffer addr (in user's mem space)
17640 000118BB BE[322F0100]       <1>     mov     esi, Cmd_Path  ; 'PATH' address
17641 000118C0 0FB6CA             <1>     movzx   ecx, dl
17642                              <1>     ;sub    cl, 1 ; 0 -> 255, 1 -> 0
17643                              <1>     ;adc    cx, 1 ; 255 -> 256, 0 -> 1
17644                              <1>     ; 30/07/2022
17645 000118C3 FEC9               <1>     dec     cl ; 0 -> 255, 1 -> 0
17646 000118C5 41                 <1>     inc     ecx ; 255 -> 256, 0 -> 1
17647                              <1>     ; EDI = Output buffer
17648                              <1>     ; CX = Buffer length
17649                              <1>     ; AL = 0 -> use ASCIIZ word in [ESI]
17650                              <1>     ; ESI = 'PATH' address (with zero tail)
17651 000118C6 E8DC81FFFF         <1>     call    get_environment_string
17652 000118CB 72D0               <1>     jc      short syspath_err
17653 000118CD 89DF               <1>     mov     edi, ebx ; User's buffer address
17654 000118CF 89E6               <1>     mov     esi, esp
17655                              <1>     ; EDI = User's buffer address
17656                              <1>     ; ECX = transfer (byte) count
17657 000118D1 E814F2FFFF         <1>     call    transfer_to_user_buffer
17658 000118D6 72C5               <1>     jc      short syspath_err
17659 000118D8 890D[348E0100]     <1>     mov     [u.r0], ecx
17660 000118DE E99BB1FFFF         <1>     jmp     sysret
17661                              <1>
17662                              <1> syspath_0:
17663 000118E3 89DE               <1>     mov     esi, ebx
17664 000118E5 0FB6CA             <1>     movzx   ecx, dl
17665                              <1>     ;sub    cl, 1 ; 0 -> 255, 1 -> 0
17666                              <1>     ;adc    cx, 1 ; 255 -> 256, 0 -> 1
17667                              <1>     ; 30/07/2022
17668 000118E8 FEC9               <1>     dec     cl ; 0 -> 255, 1 -> 0
17669 000118EA 41                 <1>     inc     ecx ; 255 -> 256, 0 -> 1
17670 000118EB E844F2FFFF         <1>     call    transfer_from_user_buffer
17671 000118F0 72AB               <1>     jc      short syspath_err
17672                              <1>     ;(*) 'PATH=' will be added to
17673                              <1>     ;         the head of the string
17674 000118F2 83EC08             <1>     sub     esp, 8 ;(*)
17675 000118F5 89FE               <1>     mov     esi, edi ;(*)
17676 000118F7 E88F81FFFF         <1>     call    set_path_x ;(*)
17677 000118FC 729F               <1>     jc      short syspath_err
17678 000118FE 8915[348E0100]     <1>     mov     [u.r0], edx ; run path string length
17679 00011904 E975B1FFFF         <1>     jmp     sysret
17680                              <1>
17681                              <1> sysenv:
17682                              <1>     ; Get/Set Environment Variables
17683                              <1>     ;
```

```
17684                                  <1>     ; 30/07/2022
17685                                  <1>     ; 23/07/2022 - TRDOS 386 v2.0.5
17686                                  <1>     ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
17687                                  <1>     ;
17688                                  <1>       ; INPUT ->
17689                                  <1>     ;       EBX = 0 -> get (all) environment variables
17690                                  <1>     ;             (Required Buffer length = 512 bytes)
17691                                  <1>     ;       EBX > 0 -> set (one) environment variable
17692                                  <1>     ;             (If there is not a '=' after
17693                                  <1>     ;             the environment variable name, it will
17694                                  <1>     ;             accepted as 'get environment variable'.)
17695                                  <1>     ;              EBX = Buffer address
17696                                  <1>     ;       ECX = Buffer address (if EBX = 0)
17697                                  <1>     ;             (ECX will not be used if EBX > 0)
17698                                  <1>     ;             (Note: Buffer size is 512 bytes.)
17699                                  <1>     ;       DL = Buffer size (0 = 256 byte)
17700                                  <1>     ;            (For one envrionment variable)
17701                                  <1>     ;
17702                                  <1>     ; OUTPUT ->
17703                                  <1>     ;       (EAX = 0)
17704                                  <1>     ;       CF = 0 -> Successful (EAX = String length)
17705                                  <1>     ;       CF = 1 -> Failed (EAX = 0)
17706                                  <1>     ;
17707                                  <1>     ; Note: Environment variable name, for example,
17708                                  <1>     ;       'PATH=' must be included at the beginning
17709                                  <1>     ;       of the environment string. If the variable
17710                                  <1>     ;       name is as 'PATH' but it is not as 'PATH='
17711                                  <1>     ;       the variable string (row) will be returned.
17712                                  <1>     ;       If variable name is as 'PATH=' but there is
17713                                  <1>     ;       not a following text after the variable name,
17714                                  <1>     ;       the environment variable will be reset/deleted.
17715                                  <1>
17716 00011909 89E5                   <1>     mov     ebp, esp
17717 0001190B 81EC00020000           <1>     sub     esp, 512
17718 00011911 89E7                   <1>     mov     edi, esp
17719                                  <1>
17720 00011913 31C0                   <1>     xor     eax, eax
17721 00011915 A3[348E0100]           <1>     mov     [u.r0], eax
17722                                  <1>
17723 0001191A 21DB                   <1>     and     ebx, ebx
17724 0001191C 751F                   <1>     jnz     short sysenv_0
17725                                  <1>
17726                                  <1>     ; EBX = 0 -> get (all) environment variables
17727 0001191E 89EC                   <1>     mov     esp, ebp
17728 00011920 BE00300900             <1>     mov     esi, Env_Page  ; Environment page
17729 00011925 89CF                   <1>     mov     edi, ecx ; buffer addr (in user's mem space)
17730                                  <1>     ;mov    ecx, 512
17731                                  <1>     ; 30/07/2022
17732 00011927 29C9                   <1>     sub     ecx, ecx
17733 00011929 B502                   <1>     mov     ch, 2
17734                                  <1>     ; ecx = 512
17735 0001192B E8BAF1FFFF             <1>     call    transfer_to_user_buffer
17736                                  <1>     ;jc     error
17737                                  <1>     ; 23/07/2022
17738 00011930 725B                   <1>     jc      short sysenv_error
17739 00011932 890D[348E0100]         <1>     mov     [u.r0], ecx
17740 00011938 E941B1FFFF             <1>     jmp     sysret
17741                                  <1>
17742                                  <1> sysenv_0:
17743 0001193D 89DE                   <1>     mov     esi, ebx ; * ; user's buffer address
17744 0001193F 0FB6CA                 <1>     movzx   ecx, dl
17745                                  <1>     ;sub    cl, 1 ; 0 -> 255, 1 -> 0
17746                                  <1>     ;adc    cx, 1 ; 255 -> 256, 0 -> 1
17747                                  <1>     ; 30/07/2022
17748 00011942 FEC9                   <1>     dec     cl ; 0 -> 255, 1 -> 0
17749 00011944 41                     <1>     inc     ecx ; 255 -> 256, 0 -> 1
17750 00011945 E8EAF1FFFF             <1>     call    transfer_from_user_buffer
17751 0001194A 723F                   <1>     jc      short sysenv_err
17752 0001194C 89FE                   <1>     mov     esi, edi
17753 0001194E 8A06                   <1>     mov     al, [esi]
17754 00011950 3C20                   <1>     cmp     al, 20h
17755 00011952 7637                   <1>     jna     short sysenv_err
17756 00011954 3C3D                   <1>     cmp     al, '='
17757 00011956 7433                   <1>     je      short sysenv_err
17758 00011958 56                     <1>     push    esi
17759                                  <1> sysenv_1:
17760 00011959 46                     <1>     inc     esi
17761 0001195A 803E3D                 <1>     cmp     byte [esi], '='
17762 0001195D 7433                   <1>     je      short sysenv_3
17763 0001195F 803E20                 <1>     cmp     byte [esi], 20h
17764 00011962 73F5                   <1>     jnb     short sysenv_1
17765 00011964 C60600                 <1>     mov     byte [esi], 0
17766 00011967 5E                     <1>     pop     esi
17767                                  <1>     ; EDI = Output buffer
17768                                  <1>     ; CX = Buffer length
17769 00011968 30C0                   <1>     xor     al, al
17770                                  <1>     ; AL = 0 -> use ASCIIZ word in [ESI]
17771                                  <1>     ; ESI = Environment variable name address
17772 0001196A E83881FFFF             <1>     call    get_environment_string
17773 0001196F 721A                   <1>     jc      short sysenv_err
17774 00011971 89DF                   <1>     mov     edi, ebx ; * ; user's buffer address
17775 00011973 89C1                   <1>     mov     ecx, eax ; String length
17776 00011975 89E6                   <1>     mov     esi, esp
17777                                  <1>     ; ESI = system buffer address
17778                                  <1>     ; EDI = User's buffer address
17779                                  <1>     ; ECX = transfer (byte) count
17780 00011977 E86EF1FFFF             <1>     call    transfer_to_user_buffer
17781 0001197C 720D                   <1>     jc      short sysenv_err
17782 0001197E 890D[348E0100]         <1>     mov     [u.r0], ecx ; transfer (byte) count
17783                                  <1> sysenv_2:
17784 00011984 89EC                   <1>     mov     esp, ebp
17785 00011986 E9F3B0FFFF             <1>     jmp     sysret
17786                                  <1> sysenv_err:
17787 0001198B 89EC                   <1>     mov     esp, ebp
17788                                  <1> sysenv_error:       ; 23/07/2022
17789 0001198D E9CCB0FFFF             <1>     jmp     error
17790                                  <1> sysenv_3:
17791 00011992 46                     <1>     inc     esi
17792 00011993 803E20                 <1>     cmp     byte [esi], 20h
17793 00011996 73FA                   <1>     jnb     short sysenv_3
17794 00011998 C60600                 <1>     mov     byte [esi], 0
17795 0001199B 5E                     <1>     pop     esi
17796 0001199C E8C981FFFF             <1>     call    set_environment_string
17797 000119A1 72E8                   <1>     jc      short sysenv_err
17798 000119A3 8915[348E0100]         <1>     mov     [u.r0], edx
17799 000119A9 EBD9                   <1>     jmp     short sysenv_2
17800                                  <1>
17801                                  <1> isintr:
17802                                  <1>     ; 21/08/2024 - TRDOS 386 v2.0.9
17803                                  <1>     ; check terminate (CTRL+BREAK)
17804                                  <1>     ;       interrupt is enabled or disabled
17805                                  <1>     ;test   word [u.intr], 0FFFFh
17806 000119AB F605[808E0100]FF       <1>     test    byte [u.intr], 0FFh
17807                                  <1>     ; zf = 1 -> terminate interrupt disabled
```

```
17808                                   <1>     ; zf = 0 -> terminate interrupt enabled
17809                                   <1>
17810                                   <1> ; 19/12/2025
17811                                   <1> iget:
17812                                   <1> sndc:
17813                                   <1> sleep:
17814 000119B2 C3                      <1>     retn
17815                                   <1>
17816                                   <1> ; 23/07/2022 - TRDOS 386 v2.0.5
17817                                   <1>
17818                                   <1> ; 22/01/2021
17819                                   <1> ; temporary - 24/01/2016
17820                                   <1>
17821                                   <1> ; 19/12/2025
17822                                   <1> ;iget:
17823                                   <1> ;   ;retn
17824                                   <1> ;;iopen:
17825                                   <1> ;;  ;retn
17826                                   <1> ;;iclose:
17827                                   <1> ;;  ;retn
17828                                   <1> ;sndc:
17829                                   <1> ;   ;retn
17830                                   <1> ;access:
17831                                   <1> ;   ;retn
17832                                   <1> ;sleep:
17833                                   <1> ;   retn
 3438                                   %include 'trdosk7.s' ; 24/01/2016
    1                                   <1> ; ****************************************************************************
    2                                   <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.6) - DISK READ&WRITE : trdosk7.s
    3                                   <1> ; ----------------------------------------------------------------------------
    4                                   <1> ; Last Update: 29/08/2023  (Previous: 25/07/2022 - Kernel v2.0.5)
    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 000119B3 807E0500                 <1>     cmp     byte [esi+LD_LBAYes], 0
   20 000119B7 7774                     <1>         ja      short lba_write
   21                                   <1>
   22                                   <1> chs_write:
   23                                   <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
   24                                   <1>     ; 25/02/2016
   25                                   <1>     ; 23/02/2016
   26 000119B9 C605[317F0100]03         <1>     mov     byte [disk_rw_op], 3 ; CHS write
   27 000119C0 EB0D                     <1>     jmp     short chs_rw
   28                                   <1>
   29                                   <1> disk_read:
   30                                   <1>     ; 25/02/2016
   31                                   <1>     ; 24/02/2016
   32                                   <1>     ; 23/02/2016
   33                                   <1>     ; 17/02/2016
   34                                   <1>     ; 14/02/2016
   35                                   <1>     ; 31/01/2016 (TRDOS 386 = TRDOS v2.0)
   36                                   <1>     ; 17/10/2010
   37                                   <1>     ; 18/04/2010
   38                                   <1>     ;
   39                                   <1>     ; INPUT -> EAX = Logical Block Address
   40                                   <1>     ;          ESI = Logical Dos Disk Table Offset (DRV)
   41                                   <1>     ;          ECX = Sector Count
   42                                   <1>     ;          EBX = Destination Buffer
   43                                   <1>     ; OUTPUT ->
   44                                   <1>     ;          cf = 0 or cf = 1
   45                                   <1>     ; (Modified registers: EAX, EBX, ECX, EDX)
   46                                   <1>
   47 000119C2 807E0500                 <1>     cmp     byte [esi+LD_LBAYes], 0
   48 000119C6 776E                     <1>         ja      short lba_read
   49                                   <1>
   50                                   <1> chs_read:
   51                                   <1>     ; 29/08/2023 (TRDOS 386 Kernel v2.0.6)
   52                                   <1>     ; 25/07/2022 (TRDOS 386 Kernel v2.0.5)
   53                                   <1>     ; 25/02/2016
   54                                   <1>     ; 24/02/2016
   55                                   <1>     ; 23/02/2016
   56                                   <1>     ; 31/01/2016 (TRDOS 386 = TRDOS v2.0)
   57                                   <1>     ; 20/07/2011
   58                                   <1>     ; 04/07/2009
   59                                   <1>     ;
   60                                   <1>     ; INPUT -> EAX = Logical Block Address
   61                                   <1>     ;          ECX = Number of sectors to read
   62                                   <1>     ;          ESI = Logical Dos Disk Table Offset (DRV)
   63                                   <1>     ;          EBX = Destination Buffer
   64                                   <1>     ; OUTPUT ->
   65                                   <1>     ;          cf = 0 or cf = 1
   66                                   <1>     ; (Modified registers: EAX, EBX, ECX, EDX)
   67                                   <1>
   68                                   <1>     ; 23/02/2016
   69 000119C8 C605[317F0100]02         <1>     mov     byte [disk_rw_op], 2 ; CHS read
   70                                   <1>
   71                                   <1> chs_rw:
   72                                   <1>     ;;movzx edx, word [esi+LD_BPB+SecPerTrack]
   73                                   <1>     ;movzx  edx, byte [esi+LD_BPB+SecPerTrack] ; <= 63
   74                                   <1>     ;mov    [disk_rw_spt], dl
   75                                   <1>
   76                                   <1> chs_read_next_sector:
   77 000119CF C605[327F0100]04         <1>     mov     byte [retry_count], 4
   78                                   <1>
   79                                   <1> chs_read_retry:
   80                                   <1>     ;mov    [sector_count], ecx ; 23/02/2016
   81                                   <1>
   82 000119D6 50                       <1>     push    eax                     ; Linear sector #
   83 000119D7 51                       <1>     push    ecx                     ; # of FAT/FILE/DIR sectors
   84                                   <1>
   85 000119D8 0FB74E1E                 <1>     movzx   ecx, word [esi+LD_BPB+SecPerTrack]
   86                                   <1>     ;movzx  ecx, byte [disk_rw_spt] ; 23/02/2016
   87 000119DC 29D2                     <1>     sub     edx, edx
   88 000119DE F7F1                     <1>     div     ecx
   89                                   <1>     ; eax = track, dx (dl ) = sector (on track)
   90                                   <1>     ;sub    cl, dl ; 24/02/2016 (spt - sec)
   91                                   <1>     ;push   ecx ; *
   92                                   <1>     ; 25/07/2022
   93                                   <1>     ;mov    cx, dx                  ; Sector (zero based)
   94                                   <1>     ;inc    cx                      ; To make it 1 based
   95                                   <1>     ;push   cx
   96                                   <1>     ; 29/08/2023
   97                                   <1>     ;mov    ecx, edx
```

```
                            <1>     ;inc    ecx
                            <1>     ;push   ecx
000119E0 42                 <1>     inc     edx
000119E1 52                 <1>     push    edx ; dl = sector number (on track)
                            <1>     ;
000119E2 668B4E20           <1>     mov     cx, [esi+LD_BPB+Heads]
                            <1>     ;sub    dx, dx
                            <1>     ; 25/07/2022
000119E6 29D2               <1>     sub     edx, edx
000119E8 F7F1               <1>     div     ecx                     ; Convert track to head & cyl
                            <1>     ; eax (ax) = cylinder, dx (dl) = head (max. FFh)
000119EA 88D6               <1>     mov     dh, dl
                            <1>     ;pop    cx                      ; AX=Cyl, DH=Head, CX=Sector
                            <1>     ; 25/07/2022
000119EC 59                 <1>     pop     ecx ; sector number (on track)
000119ED 8A5602             <1>     mov     dl, [esi+LD_PhyDrvNo]
                            <1>
000119F0 88C5               <1>     mov     ch, al                  ; NOTE: max. 1023 cylinders !
000119F2 C0CC02             <1>     ror     ah, 2                   ; Rotate 2 bits right
000119F5 08E1               <1>     or      cl, ah
                            <1>
                            <1>     ; 24/02/2016
                            <1>     ;pop    eax ; * (spt - sec) (example: 63 - 0 = 63)
                            <1>     ;cmp    eax, [sector_count]
                            <1>     ;jb     short chs_write_sectors
                            <1>     ;je     short chs_read_sectors
                            <1>     ;; (# of sectors to read is more than remaining sectors on the track)
                            <1>     ;mov    al, [sector_count]
                            <1> ;chs_read_sectors: ; read or write !
000119F7 B001               <1>     mov     al, 1 ; 25/02/2016
000119F9 8A25[317F0100]     <1>     mov     ah, [disk_rw_op]  ; 02h = chs read, 03h = chs write
                            <1>     ;
000119FF E80736FFFF         <1>     call    int13h                  ; BIOS Service func ( ah ) = 2
                            <1>                                     ; Read disk sectors
                            <1>                                     ; AL-sec num CH-track CL-sec
                            <1>                                     ; DH-head DL-drive ES:BX-buffer
                            <1>                                     ; CF-flag AH-stat AL-sec read
                            <1>                                   ; If CF = 1 then (If AH > 0)
00011A04 8825[337F0100]     <1>     mov     [disk_rw_err], ah
                            <1>
00011A0A 59                 <1>     pop     ecx
00011A0B 58                 <1>     pop     eax
00011A0C 7314               <1>     jnc     short chs_read_ok
                            <1>
00011A0E 803D[337F0100]09   <1>     cmp     byte [disk_rw_err], 09h ; DMA crossed 64K segment boundary
00011A15 7408               <1>     je      short chs_read_error_retn
                            <1>
00011A17 FE0D[327F0100]     <1>     dec     byte [retry_count]
00011A1D 75B7               <1>     jnz     short chs_read_retry
                            <1>
                            <1> chs_read_error_retn:
00011A1F F9                 <1>     stc
                            <1>     ;retn
00011A20 EB67               <1>     jmp     short update_drv_error_byte
                            <1>
                            <1> ;chs_write_sectors: ; read or write
                            <1>     ;; (# of sectors to read is less than remaining sectors on the track)
                            <1>     ;mov    [sector_count], al
                            <1>     ;jmp    short chs_read_sectors
                            <1>
                            <1> chs_read_ok:
                            <1>     ;; 23/02/2016
                            <1>     ;movzx  edx, byte [sector_count] ; sector count (<= spt)
                            <1>        ;sub    ecx, edx  ; remaining sector count
                            <1>     ;jna    short update_drv_error_byte
                            <1>     ;add    eax, edx ; next disk sector
                            <1>     ;shl    edx, 9 ; 512 * sector count
                            <1>     ;add    ebx, edx ; next buffer byte address
                            <1>        ;jmp       chs_read_next_sector
                            <1>     ; 25/02/2016
00011A22 40                 <1>     inc     eax ; next sector
00011A23 81C300020000       <1>     add     ebx, 512
00011A29 E2A4               <1>     loop    chs_read_next_sector
00011A2B EB5C               <1>     jmp     short update_drv_error_byte
                            <1>
                            <1> lba_write:
                            <1>     ; 23/02/2016
00011A2D C605[317F0100]1C   <1>     mov     byte [disk_rw_op], 1Ch ; LBA write
00011A34 EB07               <1>     jmp     short lba_rw
                            <1>
                            <1> lba_read:
                            <1>     ; 14/07/2022
                            <1>     ; 23/02/2016
                            <1>     ; 17/02/2016
                            <1>     ; 14/02/2016
                            <1>     ; 13/02/2016
                            <1>     ; 31/01/2016 (TRDOS 386 = TRDOS v2.0)
                            <1>     ; 10/07/2015 (Retro UNIX 386 v1)
                            <1>     ;
                            <1>     ; INPUT -> EAX = Logical Block Address
                            <1>     ;          ESI = Logical Dos Disk Table Offset (DRV)
                            <1>     ;          ECX = Sector Count
                            <1>     ;          EBX = Destination Buffer
                            <1>     ; OUTPUT ->
                            <1>     ;          cf = 0 or cf = 1
                            <1>     ; (Modified registers: EAX, EBX, ECX, EDX)
                            <1>
                            <1>     ; LBA read/write (with private LBA function)
                            <1>     ;((Retro UNIX 386 v1 - DISK I/O code by Erdogan Tan))
                            <1>
                            <1>     ; 23/02/2016
00011A36 C605[317F0100]1B   <1>     mov     byte [disk_rw_op], 1Bh ; LBA read
                            <1>
                            <1> lba_rw:
                            <1>     ; 17/02/2016
00011A3D 57                 <1>     push    edi
                            <1>
00011A3E 890D[347F0100]     <1>     mov     [sector_count], ecx ; total sector (read) count
                            <1>
00011A44 8A5602             <1>     mov     dl, [esi+LD_PhyDrvNo]
                            <1>     ; dl = physical drive number (0, 1, 80h, 81h, 82h, 83h)
                            <1>
                            <1> lba_read_next:
                            <1>     ; 14/07/2022
00011A47 BF00010000         <1>     mov     edi, 256
00011A4C 39F9               <1>     cmp     ecx, edi
                            <1>     ;cmp    ecx, 256
00011A4E 7602               <1>     jna     short lba_read_rsc
                            <1>     ;mov    ecx, 256 ; 17/02/2016
00011A50 89F9               <1>     mov     ecx, edi
                            <1> lba_read_rsc:
00011A52 290D[347F0100]     <1>     sub     [sector_count], ecx ; remain sectors
                            <1>
00011A58 89CF               <1>     mov     edi, ecx
```

```
222 00011A5A 89C1                <1>     mov     ecx, eax ; sector number/address
223                               <1>
224 00011A5C C605[327F0100]04     <1>     mov     byte [retry_count], 4
225                               <1> lba_read_retry:
226 00011A63 89F8                 <1>     mov     eax, edi
227                               <1>     ;
228                               <1>     ; ecx = sector number
229                               <1>     ; al = sector count (0 - 255) /// (0 = 256)
230                               <1>     ; dl = drive number
231                               <1>     ; ebx = buffer offset
232                               <1>     ;
233                               <1>     ; Function 1Bh = LBA read, 1Ch = LBA write
234                               <1>     ; 23/02/2016
235 00011A65 8A25[317F0100]       <1>     mov     ah, [disk_rw_op] ; 1Bh = LBA read, 1Ch = LBA write
236 00011A6B E89B35FFFF           <1>     call    int13h
237                               <1>     ; al = ? (changed)
238                               <1>     ; ah = error code
239 00011A70 8825[337F0100]       <1>     mov     [disk_rw_err], ah
240 00011A76 7332                 <1>     jnc     short lba_read_ok
241 00011A78 80FC80               <1>     cmp     ah, 80h ; time out?
242 00011A7B 740A                 <1>         je      short lba_read_stc_retn
243 00011A7D FE0D[327F0100]       <1>     dec     byte [retry_count]
244 00011A83 7FDE                 <1>     jg      short lba_read_retry
245 00011A85 7438                 <1>     jz      short lba_read_reset
246                               <1>     ; sf = 1
247                               <1>
248                               <1> lba_read_stc_retn:
249 00011A87 F9                   <1>     stc
250                               <1> lba_read_retn:
251 00011A88 5F                   <1>     pop     edi
252                               <1>
253                               <1> update_drv_error_byte:
254 00011A89 9C                   <1>     pushf
255 00011A8A 53                   <1>     push    ebx
256                               <1>     ;push   cx
257 00011A8B 51                   <1>     push    ecx ; 14/07/2022
258                               <1>     ;or     ecx, ecx
259                               <1>     ;jz     short udrv_errb0
260 00011A8C 8A0D[337F0100]       <1>     mov     cl, [disk_rw_err]
261                               <1> udrv_errb0:
262 00011A92 0FB65E02             <1>     movzx   ebx, byte [esi+LD_PhyDrvNo]
263 00011A96 80FB02               <1>     cmp     bl, 2
264 00011A99 7203                 <1>         jb      short udrv_errb1
265 00011A9B 80EB7E               <1>     sub     bl, 7Eh
266                               <1>     ;cmp    bl, 5
267                               <1>     ;ja     short udrv_errb2
268                               <1> udrv_errb1:
269 00011A9E 81C3[A5660000]       <1>         add     ebx, drv.error ; 13/02/2016
270 00011AA4 880B                 <1>     mov     [ebx], cl ; error code
271                               <1> udrv_errb2:
272                               <1>     ;pop    cx
273 00011AA6 59                   <1>     pop     ecx  ; 14/07/2022
274 00011AA7 5B                   <1>     pop     ebx
275 00011AA8 9D                   <1>     popf
276 00011AA9 C3                   <1>     retn
277                               <1>
278                               <1> lba_read_ok:
279 00011AAA 89C8                 <1>     mov     eax, ecx ; sector number
280 00011AAC 01F8                 <1>     add     eax, edi ; sector number (next)
281 00011AAE C1E709               <1>     shl     edi, 9 ; sector count * 512
282 00011AB1 01FB                 <1>     add     ebx, edi ; next buffer offset
283                               <1>
284 00011AB3 8B0D[347F0100]       <1>     mov     ecx, [sector_count] ; remaining sectors
285 00011AB9 09C9                 <1>     or      ecx, ecx
286 00011ABB 758A                 <1>     jnz     short lba_read_next
287 00011ABD EBC9                 <1>     jmp     short lba_read_retn
288                               <1>
289                               <1> lba_read_reset:
290 00011ABF B40D                 <1>     mov     ah, 0Dh ; Alternate reset
291 00011AC1 E84535FFFF           <1>         call        int13h
292                               <1>     ; al = ? (changed)
293                               <1>     ; ah = error code
294 00011AC6 739B                 <1>     jnc     short lba_read_retry
295 00011AC8 EBBE                 <1>     jmp     short lba_read_retn
3439                                  %include 'trdosk8.s' ; 24/01/2016
   1                              <1> ; ****************************************************************************
   2                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.10) - MAIN PROGRAM : trdosk8.s
   3                              <1> ; ----------------------------------------------------------------------------
   4                              <1> ; Last Update: 28/01/2025  (Previous: 29/12/2024)
   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> ; u0.s (20/11/2015), u4.s (14/10/2015)
  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> ; DIR.ASM (c) 2004-2011 Erdogan TAN  [07/01/2004] Last Update: 09/10/2011
  17                              <1>
  18                              <1> set_run_sequence:
  19                              <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
  20                              <1>     ; 23/12/2016
  21                              <1>     ; 10/06/2016
  22                              <1>     ; 22/05/2016
  23                              <1>     ; 20/05/2016
  24                              <1>     ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
  25                              <1>     ; TRDOS 386 feature only !
  26                              <1>     ;
  27                              <1>     ; INPUT ->
  28                              <1>     ;       AL = process number (next process)
  29                              <1>     ;
  30                              <1>     ;       this process must be added to run sequence
  31                              <1>     ;
  32                              <1>     ;       [u.pri] = priority of present process
  33                              <1>     ;
  34                              <1>     ;       DL = priority (queue)
  35                              <1>     ;            0 = background (low) ; run on background
  36                              <1>     ;            1 = regular (normal) ; run as regular
  37                              <1>     ;            2 = event (high)     ; run for event
  38                              <1>     ;
  39                              <1>     ;       1) If the requested process is already running:
  40                              <1>     ;          a) If present priority is high ([u.pri]=2)
  41                              <1>     ;             and requested priority is also high,
  42                              <1>     ;             there is nothing to do! Because it has been
  43                              <1>     ;             done already (before this attempt).
  44                              <1>     ;          b) If present priority is high ([u.pri]=2)
  45                              <1>     ;             and requested priority is not high, there is
  46                              <1>     ;             nothing to do! Because, it's current
  47                              <1>     ;             run queue is unspecified, here. (It may be in
  48                              <1>     ;             a waiting list or in a run queue; if the new
  49                              <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:
00011ACA 3A05[858E0100] <1>         cmp     al, [u.uno]     ; same process ?
00011AD0 750C           <1>     jne     short srunseq_2 ; no
                        <1>
00011AD2 8A25[7E8E0100] <1>     mov     ah, [u.pri]     ; present/current priority
00011AD8 80FC02         <1>     cmp     ah, 2           ; 'run for event' priority level
00011ADB 7220           <1>     jb      short srunseq_6 ; no
                        <1>
                        <1> srunseq_1:
                        <1>     ; there is nothing to do!
00011ADD 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
00011ADE 80FA02         <1>     cmp     dl, 2           ; event queue
00011AE1 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
00011AE3 3A15[7E8E0100] <1>     cmp     dl, [u.pri]     ; priority of present process
00011AE9 7606           <1>     jna     short srunseq_3 ; is high, also
                        <1>     ;
                        <1>     ; present process will be swapped/switched out
00011AEB FE05[11830100] <1>     inc     byte [p_change] ; 1
                        <1>
                        <1> srunseq_3:
                        <1>     ; add process to 'runq_event' queue for new event
00011AF1 BB[228E0100]   <1>     mov     ebx, runq_event ; high priority run queue
                        <1>
                        <1> srunseq_4:
                        <1>     ; al = process number
                        <1>     ; ebx = run queue
                        <1>     ;call   putlu
                        <1>     ;retn
                        <1>     ; 29/07/2022
00011AF6 E91BEFFFFF     <1>     jmp     putlu
                        <1>
                        <1> srunseq_5:
00011AFB F5             <1>     cmc
00011AFC C3             <1>     retn
                        <1>
                        <1> srunseq_6:
                        <1>     ; present priority of the process is not high
                        <1>
00011AFD 8815[7E8E0100] <1>     mov     [u.pri], dl ; new priority
                        <1>                         ; (will be used by 'tswap')
                        <1>
00011B03 80FA02         <1>     cmp     dl, 2           ; high priority ?
00011B06 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>
00011B08 BB[248E0100]           <1>     mov     ebx, runq_normal ; 'run as regular' queue
                                <1>
00011B0D 20E4                   <1>     and     ah, ah  ;  previous value of [u.pri]
00011B0F 75E5                   <1>     jnz     short srunseq_4
                                <1>
00011B11 43                     <1>     inc     ebx
00011B12 43                     <1>     inc     ebx
                                <1>     ; ebx = runq_background ; 'run on backgroud' queue
                                <1>
00011B13 EBE1                   <1>     jmp     short srunseq_4
                                <1> clock:
                                <1>     ; 21/08/2024 - TRDOS 386 v2.0.9
                                <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>
                                <1>     ;;;
                                <1>     ; 21/08/2024
                                <1>     ;cmp    word [u.quit], 0FFFFh
00011B15 803D[828E0100]FF       <1>     cmp     byte [u.quit], 0FFh
00011B1C 721A                   <1>     jb      short clk_0
                                <1>     ; CTRL+BRK keys pressed
                                <1>     ;cmp    word [u.intr], 0
00011B1E 803D[808E0100]00       <1>     cmp     byte [u.intr], 0
00011B25 7611                   <1>     jna     short clk_0
00011B27 C605[11830100]FF       <1>     mov     byte [p_change], 0FFh ; -1 ; CTRL+BREAK sign
00011B2E F605[7C8E0100]FF       <1>     test    byte [u.quant], 0FFh  ; if [u.quant] number > 0
00011B35 752D                   <1>     jnz     short clk_1 ; decrease [u.quant] number
                                <1>     ; [u.quant] = 0
00011B37 C3                     <1>     retn
                                <1>     ;;;
                                <1> clk_0:
00011B38 803D[7C8E0100]00       <1>     cmp     byte [u.quant], 0
00011B3F 7723                   <1>     ja      short clk_1
                                <1>     ;
00011B41 803D[858E0100]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.
00011B48 761A                   <1>     jna     short clk_1 ; yes, do not swap out
                                <1>     ;
00011B4A 803D[288E0100]FF       <1>     cmp     byte [sysflg], 0FFh ; user or system space ?
00011B51 7517                   <1>     jne     short clk_2         ; system space (sysflg <> 0FFh)
                                <1>     ;
                                <1>     ; 21/08/2024
                                <1>     ;cmp    word [u.intr], 0
                                <1>     ;jna    short clk_2
                                <1>     ;
                                <1>     ; 23/05/2016
00011B53 803D[12830100]00       <1>     cmp     byte [multi_tasking], 0
00011B5A 760E                   <1>     jna     short clk_2
                                <1>
                                <1>     ; 21/08/2024
                                <1>     ;inc    byte [p_change] ; it is time to change running process
00011B5C C605[11830100]01       <1>     mov     byte [p_change], 1
00011B63 C3                     <1>     retn
                                <1> clk_1:
00011B64 FE0D[7C8E0100]         <1>     dec     byte [u.quant]
                                <1> clk_2:
00011B6A 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>     ;
                                <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
00011B6B 80FC07                 <1>     cmp     ah, 7
00011B6E 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
00011B70 FB                     <1>     sti
                                <1>
                                <1>     ;sti    ; enable interrupts
00011B71 80642408FE             <1>     and     byte [esp+8], 11111110b       ; clear carry bit of eflags register
                                <1>
00011B76 80FC01                 <1>     cmp     ah, 1
00011B79 7205                   <1>     jb      short int34h_0
00011B7B 7705                   <1>     ja      short int34h_1
```

```
                        <1>
00011B7D EE             <1>     out     dx, al
                        <1>     ;iretd
00011B7E EB01           <1>     jmp     short int34h_iret
                        <1>
                        <1> int34h_0:
00011B80 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
00011B81 CF             <1>     iretd
                        <1>
                        <1> int34h_1:
00011B82 F6C401         <1>     test    ah, 1
00011B85 7516           <1>     jnz     short int34h_3 ; out
                        <1>
                        <1>     ; in
00011B87 80FC02         <1>     cmp     ah, 2
00011B8A 7707           <1>     ja      short int34h_2
                        <1>
00011B8C 6689D8         <1>     mov     ax, bx
00011B8F 66ED           <1>     in      ax, dx
                        <1>     ;iretd
00011B91 EBEE           <1>     jmp     short int34h_iret
                        <1>
                        <1> int34h_2:
00011B93 80FC04         <1>     cmp     ah, 4
00011B96 772C           <1>     ja      short int34h_7 ; 12/10/2017
                        <1>     ; ah = 4
00011B98 89D8           <1>     mov     eax, ebx
00011B9A ED             <1>     in      eax, dx
                        <1>     ;iretd
00011B9B EBE4           <1>     jmp     short int34h_iret
                        <1>
                        <1> int34h_3:
00011B9D 80FC03         <1>     cmp     ah, 3
00011BA0 7707           <1>     ja      short int34h_4
                        <1>
00011BA2 6689D8         <1>     mov     ax, bx
00011BA5 66EF           <1>     out     dx, ax
                        <1>     ;iretd
00011BA7 EBD8           <1>     jmp     short int34h_iret
                        <1>
                        <1> int34h_4:
00011BA9 80FC05         <1>     cmp     ah, 5
00011BAC 770B           <1>     ja      short int34h_6 ; 12/10/2017
                        <1>     ; ah = 5
00011BAE 89D8           <1>     mov     eax, ebx
00011BB0 EF             <1>     out     dx, eax
                        <1>     ;iretd
00011BB1 EBCE           <1>     jmp     short int34h_iret
                        <1>
                        <1> int34h_5:
00011BB3 804C240801     <1>     or      byte [esp+8], 1       ; set carry bit of eflags register
00011BB8 CF             <1>     iretd
                        <1>
                        <1>     ; 12/10/2017
                        <1> int34h_6:
00011BB9 6689D8         <1>     mov     ax, bx
00011BBC EE             <1>     out     dx, al
00011BBD EB00           <1>     jmp     short $+2
00011BBF 86C4           <1>     xchg    ah, al
00011BC1 EE             <1>     out     dx, al
                        <1>     ;xchg   al, ah
                        <1>     ;iretd
00011BC2 EB06           <1>     jmp     short int34h_8
                        <1> int34h_7:
00011BC4 EC             <1>     in      al, dx
00011BC5 EB00           <1>     jmp     short $+2
00011BC7 88C4           <1>     mov     ah, al
00011BC9 EC             <1>     in      al, dx
                        <1> int34h_8:
00011BCA 86E0           <1>     xchg    al, ah
00011BCC CF             <1>     iretd
                        <1>
                        <1> INT4Ah:
                        <1>     ; 24/01/2016
                        <1>     ; this procedure will be called by 'RTC_INT' (in 'timer.s')
00011BCD 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>     ;
00011BCE 890424         <1>     mov     [esp], eax ; overwrite call return address
                        <1>     ;push   eax
00011BD1 66B80900       <1>     mov     ax, 9
00011BD5 EB07           <1>     jmp     short comm_int
                        <1> com1_int:
                        <1>     ; 07/11/2015
                        <1>     ; 24/10/2015
00011BD7 890424         <1>     mov     [esp], eax ; overwrite call return address
                        <1>     ; 23/10/2015
                        <1>     ;push   eax
00011BDA 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
00011BDE 53             <1>     push    ebx
00011BDF 56             <1>     push    esi
00011BE0 57             <1>     push    edi
00011BE1 1E             <1>     push    ds
00011BE2 06             <1>     push    es
                        <1>     ; 18/11/2015
00011BE3 0F20DB         <1>     mov     ebx, cr3
```

```
00011BE6 53                <1>     push    ebx ; ****
                           <1>     ;
00011BE7 51                <1>     push    ecx ; ***
00011BE8 52                <1>     push    edx ; **
                           <1>     ;
00011BE9 BB10000000        <1>     mov     ebx, KDATA
00011BEE 8EDB              <1>     mov     ds, bx
00011BF0 8EC3              <1>     mov     es, bx
                           <1>     ;
00011BF2 8B0D[80760100]    <1>     mov     ecx, [k_page_dir]
00011BF8 0F22D9            <1>     mov     cr3, ecx
                           <1>     ; 20/11/2015
                           <1>     ; Interrupt identification register
00011BFB 66BAFA02          <1>     mov     dx, 2FAh ; COM2
                           <1>     ;
00011BFF 3C08              <1>     cmp     al, 8
00011C01 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>     ;
00011C03 FEC6              <1>     inc     dh ; 3FAh ; COM1 Interrupt id. register
                           <1> com_i0:
                           <1>     ;push   eax ; *
                           <1>     ; 07/11/2015
00011C05 A2[EA760100]      <1>     mov     byte [ccomport], al
                           <1>     ; 09/11/2015
00011C0A 0FB7D8            <1>     movzx   ebx, ax ; 8 or 9
                           <1>     ; 17/11/2015
                           <1>     ; reset request for response status
00011C0D 88A3[E0760100]    <1>     mov     [ebx+req_resp-8], ah ; 0
                           <1>     ;
                           <1>     ; 20/11/2015
00011C13 EC                <1>     in      al, dx          ; read interrupt id. register
00011C14 EB00              <1>     JMP     $+2             ; I/O DELAY
00011C16 2404              <1>     and     al, 4           ; received data available?
00011C18 7470              <1>     jz      short com_eoi   ; (transmit. holding reg. empty)
                           <1>     ;
                           <1>     ; 20/11/2015
00011C1A 80EA02            <1>     sub     dl, 3FAh-3F8h   ; data register (3F8h, 2F8h)
00011C1D EC                <1>     in      al, dx          ; read character
                           <1>     ;JMP    $+2             ; I/O DELAY
                           <1>     ; 08/11/2015
                           <1>     ; 07/11/2015
00011C1E 89DE              <1>     mov     esi, ebx
00011C20 89DF              <1>     mov     edi, ebx
00011C22 81C6[E4760100]    <1>     add     esi, rchar - 8 ; points to last received char
00011C28 81C7[E6760100]    <1>     add     edi, schar - 8 ; points to last sent char
00011C2E 8806              <1>     mov     [esi], al ; received char (current char)
                           <1>     ; query
00011C30 20C0              <1>     and     al, al
00011C32 7527              <1>     jnz     short com_i2
                           <1>     ; response
                           <1>     ; 17/11/2015
                           <1>     ; set request for response status
00011C34 FE83[E0760100]    <1>       inc     byte [ebx+req_resp-8] ; 1
                           <1>     ;
00011C3A 6683C205          <1>     add     dx, 3FDh-3F8h   ; (3FDh, 2FDh)
00011C3E EC                <1>     in      al, dx          ; read line status register
00011C3F EB00              <1>     JMP     $+2             ; I/O DELAY
00011C41 2420              <1>     and     al, 20h         ; transmitter holding reg. empty?
00011C43 7445              <1>     jz      short com_eoi   ; no
00011C45 B0FF              <1>     mov     al, 0FFh        ; response
00011C47 6683EA05          <1>     sub     dx, 3FDh-3F8h   ; data port (3F8h, 2F8h)
00011C4B EE                <1>     out     dx, al          ; send on serial port
                           <1>     ; 17/11/2015
00011C4C 803F00            <1>     cmp     byte [edi], 0   ; query ? (schar)
00011C4F 7502              <1>     jne     short com_i1    ; no
00011C51 8807              <1>     mov     [edi], al       ; 0FFh (responded)
                           <1> com_i1:
                           <1>     ; 17/11/2015
                           <1>     ; reset request for response status (again)
00011C53 FE8B[E0760100]    <1>       dec     byte [ebx+req_resp-8] ; 0
00011C59 EB2F              <1>     jmp     short com_eoi
                           <1> com_i2:
                           <1>     ; 08/11/2015
00011C5B 3CFF              <1>     cmp     al, 0FFh        ; (response ?)
00011C5D 7417              <1>     je      short com_i3    ; (check for response signal)
                           <1>     ; 07/11/2015
00011C5F 3C04              <1>     cmp     al, 04h ; EOT
00011C61 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)
00011C63 861D[AE760100]    <1>     xchg    bl, [ptty]  ; tty number (8 or 9)
00011C69 E8C051FFFF        <1>     call    ctrlbrk
00011C6E 861D[AE760100]    <1>     xchg    [ptty], bl ; (restore ptty value and BL value)
                           <1>     ;mov    al, 04h ; EOT
                           <1>     ; 08/11/2015
00011C74 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
00011C76 803F00            <1>       cmp     byte [edi], 0 ; query ? (schar)
00011C79 7704              <1>     ja      short com_i4 ; no
                           <1>     ; reset query status (schar)
00011C7B 8807              <1>     mov     [edi], al ; 0FFh
00011C7D FEC0              <1>     inc     al ; 0
                           <1> com_i4:
                           <1>     ; 27/07/2014
                           <1>     ; 09/07/2014
00011C7F D0E3              <1>     shl     bl, 1
00011C81 81C3[B0760100]    <1>     add     ebx, ttychr
                           <1>     ; 23/07/2014 (always overwrite)
                           <1>     ;;cmp   word [ebx], 0
                           <1>     ;;ja    short com_eoi
                           <1>     ;
00011C87 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 ; *
```

```
00011C8A A0[EA760100]           <1>     mov     al, byte [ccomport] ; current COM port
                                <1>      ; al = tty number (8 or 9)
00011C8F E80F000000             <1>         call       wakeup
                                <1> com_iret:
                                <1>     ; 23/10/2015
00011C94 5A                     <1>     pop     edx ; **
00011C95 59                     <1>     pop     ecx ; ***
                                <1>     ; 18/11/2015
                                <1>     ;pop    eax ; ****
                                <1>     ;mov    cr3, eax
                                <1>     ;jmp    iiret
00011C96 E949F1FEFF             <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> ; 21/11/2023
                                <1> %if 0
                                <1>
                                <1> sp_init:
                                <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
                                <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>     ;
                                <1>     mov     ebx, com1p              ; COM1 parameters
                                <1>     mov     dx, 3F8h                ; COM1
                                <1>      ; 29/10/2015
                                <1>     mov     cx, 301h  ; divisor = 1 (115200 baud)
                                <1>     call    sp_i3   ; call A4
                                <1>     test    al, 80h
                                <1>     jz      short sp_i0 ; OK..
                                <1>             ; Error !
                                <1>     ;mov    dx, 3F8h
                                <1>     sub     dl, 5 ; 3FDh -> 3F8h
                                <1>     mov     cx, 30Eh  ; divisor = 12 (9600 baud)
                                <1>     call    sp_i3   ; call A4
                                <1>     test    al, 80h
                                <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>     ;
                                <1>     mov     byte [ebx], 0E3h ; 11100011b
                                <1>     call    sp_i5 ; 29/06/2015
                                <1> sp_i1:
                                <1>     inc     ebx
                                <1>     mov     dx, 2F8h                ; COM2
                                <1>      ; 29/10/2015
                                <1>     mov     cx, 301h  ; divisor = 1 (115200 baud)
                                <1>     call    sp_i3   ; call A4
                                <1>     test    al, 80h
                                <1>     jz      short sp_i2 ; OK..
                                <1>             ; Error !
                                <1>     ;mov    dx, 2F8h
                                <1>     sub     dl, 5 ; 2FDh -> 2F8h
                                <1>     mov     cx, 30Eh  ; divisor = 12 (9600 baud)
                                <1>     call    sp_i3   ; call A4
                                <1>     test    al, 80h
                                <1>     jnz     short sp_i7
                                <1> sp_i2:
                                <1>     mov     byte [ebx], 0E3h ; 11100011b
                                <1> sp_i6:
                                <1>     ;; COM2 - enabling IRQ 3
                                <1>     ; 29/07/2022
                                <1>     ; 07/11/2015
                                <1>     ; 26/10/2015
                                <1>     pushf
                                <1>     cli
```

```
670                             <1>     ;
671                             <1>     mov     dx, 2FCh                ; modem control register
672                             <1>     in      al, dx                  ; read register
673                             <1>     JMP     $+2                     ; I/O DELAY
674                             <1>     or      al, 8                   ; enable bit 3 (OUT2)
675                             <1>     out     dx, al                  ; write back to register
676                             <1>     JMP     $+2                     ; I/O DELAY
677                             <1>     ;mov    dx, 2F9h                ; interrupt enable register
678                             <1>     ; 29/07/2022
679                             <1>     mov     dl, 0F9h
680                             <1>     in      al, dx                  ; read register
681                             <1>     JMP     $+2                     ; I/O DELAY
682                             <1>     ;or     al, 1                   ; receiver data interrupt enable and
683                             <1>     or      al, 3                   ; transmitter empty interrupt enable
684                             <1>     out     dx, al                  ; write back to register
685                             <1>     JMP     $+2                     ; I/O DELAY
686                             <1>     in      al, 21h                 ; read interrupt mask register
687                             <1>     JMP     $+2                     ; I/O DELAY
688                             <1>     and     al, 0F7h                ; enable IRQ 3 (COM2)
689                             <1>     out     21h, al                 ; write back to register
690                             <1>     ;
691                             <1>     ; 23/10/2015
692                             <1>     mov     eax, com2_int
693                             <1>     mov     [com2_irq3], eax
694                             <1>     ; 26/10/2015
695                             <1>     popf
696                             <1> sp_i7:
697                             <1>     retn
698                             <1>
699                             <1> sp_i3:
700                             <1> ;A4:        ;----- INITIALIZE THE COMMUNICATIONS PORT
701                             <1>     ; 28/10/2015
702                             <1>     inc     dl      ; 3F9h (2F9h)  ; 3F9h, COM1 Interrupt enable register
703                             <1>     mov     al, 0
704                             <1>     out     dx, al                  ; disable serial port interrupt
705                             <1>     JMP     $+2                     ; I/O DELAY
706                             <1>     add     dl, 2   ; 3FBh (2FBh)  ; COM1 Line control register (3FBh)
707                             <1>     mov     al, 80h
708                             <1>     out     dx, al                  ; SET DLAB=1 ; divisor latch access bit
709                             <1>     ;----- SET BAUD RATE DIVISOR
710                             <1>     ; 26/10/2015
711                             <1>     sub     dl, 3   ; 3F8h (2F8h) ; register for least significant byte
712                             <1>                                     ; of the divisor value
713                             <1>     mov     al, cl  ; 1
714                             <1>     out     dx, al                  ; 1 = 115200 baud (Retro UNIX 386 v1)
715                             <1>                                     ; 2 = 57600 baud
716                             <1>                                     ; 3 = 38400 baud
717                             <1>                                     ; 6 = 19200 baud
718                             <1>                                     ; 12 = 9600 baud (Retro UNIX 8086 v1)
719                             <1>     JMP     $+2                     ; I/O DELAY
720                             <1>     sub     al, al
721                             <1>     inc     dl      ; 3F9h (2F9h) ; register for most significant byte
722                             <1>                                     ; of the divisor value
723                             <1>     out     dx, al ; 0
724                             <1>     JMP     $+2                     ; I/O DELAY
725                             <1>     ;
726                             <1>     mov     al, ch ; 3              ; 8 data bits, 1 stop bit, no parity
727                             <1>     ;and    al, 1Fh ; Bits 0,1,2,3,4
728                             <1>     add     dl, 2   ; 3FBh (2FBh)  ; Line control register
729                             <1>     out     dx, al
730                             <1>     JMP     $+2                     ; I/O DELAY
731                             <1>     ; 29/10/2015
732                             <1>     dec     dl      ; 3FAh (2FAh)  ; FIFO Control register (16550/16750)
733                             <1>     xor     al, al                  ; 0
734                             <1>     out     dx, al                  ; Disable FIFOs (reset to 8250 mode)
735                             <1>     JMP     $+2
736                             <1> sp_i4:
737                             <1> ;A18:       ;----- COMM PORT STATUS ROUTINE
738                             <1>     ; 29/06/2015 (line status after modem status)
739                             <1>     add     dl, 4   ; 3FEh (2FEh)  ; Modem status register
740                             <1> sp_i4s:
741                             <1>     in      al, dx                  ; GET MODEM CONTROL STATUS
742                             <1>     JMP     $+2                     ; I/O DELAY
743                             <1>     mov     ah, al                  ; PUT IN (AH) FOR RETURN
744                             <1>     dec     dl      ; 3FDh (2FDh)  ; POINT TO LINE STATUS REGISTER
745                             <1>                                     ; dx = 3FDh for COM1, 2FDh for COM2
746                             <1>     in      al, dx                  ; GET LINE CONTROL STATUS
747                             <1>     ; AL = Line status, AH = Modem status
748                             <1>     retn
749                             <1>
750                             <1> sp_status:
751                             <1>     ; 29/06/2015
752                             <1>     ; 27/06/2015 (Retro UNIX 386 v1)
753                             <1>     ; Get serial port status
754                             <1>     mov     dx, 3FEh                ; Modem status register (COM1)
755                             <1>     sub     dh, al                  ; dh = 2 for COM2 (al = 1)
756                             <1>                                     ; dx = 2FEh for COM2
757                             <1>     jmp     short sp_i4s
758                             <1>
759                             <1> sp_setp: ; Set serial port communication parameters
760                             <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
761                             <1>     ; 07/11/2015
762                             <1>     ; 29/10/2015
763                             <1>     ; 29/06/2015
764                             <1>     ; Retro UNIX 386 v1 feature only !
765                             <1>     ;
766                             <1>     ; INPUT:
767                             <1>     ;       AL = 0 for COM1
768                             <1>     ;            1 for COM2
769                             <1>     ;       AH = Communication parameters (*)
770                             <1>     ; OUTPUT:
771                             <1>     ;       CL = Line status
772                             <1>     ;       CH = Modem status
773                             <1>     ;   If cf = 1 -> Error code in [u.error]
774                             <1>     ;                'invalid parameter !'
775                             <1>     ;                        or
776                             <1>     ;                'device not ready !' error
777                             <1>     ;
778                             <1>     ;  (*) Communication parameters (except BAUD RATE):
779                             <1>     ;       Bit     4       3       2       1       0
780                             <1>     ;               -PARITY--   STOP BIT  -WORD LENGTH-
781                             <1>     ;  this one -->       00 = none    0 = 1 bit  11 = 8 bits
782                             <1>     ;               01 = odd     1 = 2 bits       10 = 7 bits
783                             <1>     ;               11 = even
784                             <1>     ;  Baud rate setting bits: (29/06/2015)
785                             <1>     ;               Retro UNIX 386 v1 feature only !
786                             <1>     ;       Bit     7    6    5  | Baud rate
787                             <1>     ;               ------------------------
788                             <1>     ;       value   0    0    0  | Default (Divisor = 1)
789                             <1>     ;               0    0    1  | 9600 (12)
790                             <1>     ;               0    1    0  | 19200 (6)
791                             <1>     ;               0    1    1  | 38400 (3)
792                             <1>     ;               1    0    0  | 14400 (8)
793                             <1>     ;               1    0    1  | 28800 (4)
```

```
794                                     <1>     ;             1    1    0  | 57600 (2)
795                                     <1>     ;             1    1    1  | 115200 (1)
796                                     <1>     ;
797                                     <1>     ; (COM1 base port address = 3F8h, COM1 Interrupt = IRQ 4)
798                                     <1>     ; (COM2 base port address = 2F8h, COM1 Interrupt = IRQ 3)
799                                     <1>     ;
800                                     <1>     ; ((Modified registers: EAX, ECX, EDX, EBX))
801                                     <1>     ;
802                                     <1>     mov     dx, 3F8h
803                                     <1>     mov     ebx, com1p ; COM1 control byte offset
804                                     <1>     cmp     al, 1
805                                     <1>     ja      short sp_invp_err
806                                     <1>     jb      short sp_setp1 ;  COM1 (AL = 0)
807                                     <1>     dec     dh ; 2F8h
808                                     <1>     inc     ebx ; COM2 control byte offset
809                                     <1> sp_setp1:
810                                     <1>     ; 29/10/2015
811                                     <1>     mov     [ebx], ah
812                                     <1>     movzx   ecx, ah
813                                     <1>     shr     cl, 5 ; -> baud rate index
814                                     <1>     and     ah, 1Fh ; communication parameters except baud rate
815                                     <1>     mov     al, [ecx+b_div_tbl]
816                                     <1>     mov     cx, ax
817                                     <1>     call    sp_i3
818                                     <1>     mov     cx, ax ; CL = Line status, CH = Modem status
819                                     <1>     test    al, 80h
820                                     <1>     jz      short sp_setp2
821                                     <1>         mov     byte [ebx], 0E3h ; Reset to initial value (11100011b)
822                                     <1> stp_dnr_err:
823                                     <1>     mov     dword [u.error], ERR_DEV_NOT_RDY ; 'device not ready !'
824                                     <1>     ; CL = Line status, CH = Modem status
825                                     <1>     stc
826                                     <1>     retn
827                                     <1> sp_setp2:
828                                     <1>     cmp     dh, 2 ; COM2 (2F?h)
829                                     <1>     ;jna    sp_i6
830                                     <1>                   ; COM1 (3F?h)
831                                     <1>     ; 29/07/2022
832                                     <1>     ja      short sp_i5
833                                     <1>     jmp     sp_i6
834                                     <1> sp_i5:
835                                     <1>     ; 29/07/2022
836                                     <1>     ; 07/11/2015
837                                     <1>     ; 26/10/2015
838                                     <1>     ; 29/06/2015
839                                     <1>     ;
840                                     <1>     ;; COM1 - enabling IRQ 4
841                                     <1>     pushf
842                                     <1>     cli
843                                     <1>     mov     dx, 3FCh                ; modem control register
844                                     <1>     in      al, dx                  ; read register
845                                     <1>     JMP     $+2                     ; I/O DELAY
846                                     <1>     or      al, 8                   ; enable bit 3 (OUT2)
847                                     <1>     out     dx, al                  ; write back to register
848                                     <1>     JMP     $+2                     ; I/O DELAY
849                                     <1>     ;mov    dx, 3F9h                ; interrupt enable register
850                                     <1>     ; 29/07/2022
851                                     <1>     mov     dl, 0F9h
852                                     <1>     in      al, dx                  ; read register
853                                     <1>     JMP     $+2                     ; I/O DELAY
854                                     <1>     ;or     al, 1                   ; receiver data interrupt enable and
855                                     <1>     or      al, 3                   ; transmitter empty interrupt enable
856                                     <1>     out     dx, al                  ; write back to register
857                                     <1>     JMP     $+2                     ; I/O DELAY
858                                     <1>     in      al, 21h                 ; read interrupt mask register
859                                     <1>     JMP     $+2                     ; I/O DELAY
860                                     <1>     and     al, 0EFh                ; enable IRQ 4 (COM1)
861                                     <1>     out     21h, al                 ; write back to register
862                                     <1>     ;
863                                     <1>     ; 23/10/2015
864                                     <1>     mov     eax, com1_int
865                                     <1>     mov     [com1_irq4], eax
866                                     <1>     ; 26/10/2015
867                                     <1>     popf
868                                     <1>     retn
869                                     <1>
870                                     <1> sp_invp_err:
871                                     <1>     mov     dword [u.error], ERR_INV_PARAMETER ; 'invalid parameter !'
872                                     <1>     xor     ecx, ecx
873                                     <1>     dec     ecx ; 0FFFFh
874                                     <1>     stc
875                                     <1>     retn
876                                     <1>
877                                     <1> ; 29/10/2015
878                                     <1> b_div_tbl: ; Baud rate divisor table (115200/divisor)
879                                     <1>     db 1, 12, 6, 3, 8, 4, 1
880                                     <1>
881                                     <1> %endif
882                                     <1>
883                                     <1> ; 23/10/2015
884                                     <1> com1_irq4:
885 00011C9B [A31C0100]                 <1>     dd dummy_retn
886                                     <1> com2_irq3:
887 00011C9F [A31C0100]                 <1>     dd dummy_retn
888                                     <1>
889                                     <1> ; 21/11/2023
890                                     <1> dummy_retn:
891                                     <1>     ;retn
892                                     <1> wakeup:
893                                     <1>     ; 24/01/2016
894 00011CA3 C3                         <1>     retn
895                                     <1>
896                                     <1> set_working_path_x:
897                                     <1>             ; 17/10/2016 (TRDOS 386 - FFF & FNF)
898                                     <1>             ;mov    ax, 1
899                                     <1>                     ; File name is needed/forced (AL=1)
900                                     <1>                     ; Change directory as temporary (AH=0)
901                                     <1>             ; 29/07/2022
902 00011CA4 31C0                       <1>             xor     eax, eax
903 00011CA6 FEC0                       <1>             inc     al
904                                     <1>             ; eax = 1
905                                     <1> set_working_path_xx: ; 30/12/2017 (syschdir)
906                                     <1>             ; This is needed for preventing wrong Find Next File
907                                     <1>             ; system call after sysopen, syscreate, sysmkdir etc.
908                                     <1>             ; Find Next File must immediate follow Find First File)
909                                     <1>
910 00011CA8 8825[38830100]            <1>             mov     [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
911                                     <1>
912                                     <1> set_working_path:
913                                     <1>             ; 08/08/2022
914                                     <1>             ; 29/07/2022 - TRDOS 386 Kernel v2.0.5
915                                     <1>             ; 16/10/2016
916                                     <1>             ; 12/10/2016
917                                     <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>
00011CAE 66A3[3C830100] <1>             mov     [SWP_Mode], ax
00011CB4 A0[42770100]   <1>             mov     al, [Current_Drv]
00011CB9 30E4           <1>             xor     ah, ah
00011CBB 66A3[3E830100] <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>
00011CC1 55             <1>             push    ebp
00011CC2 89E5           <1>             mov     ebp, esp
                        <1>
                        <1>             ;mov    ecx, 128 ; maximum path length = 128 bytes
                        <1>             ; 29/07/2022
00011CC4 31C9           <1>             xor     ecx, ecx
00011CC6 B180           <1>             mov     cl, 128
00011CC8 29CC           <1>             sub     esp, ecx ; reserve 128 bytes (buffer) on stack
00011CCA 89E7           <1>             mov     edi, esp ; destination address (kernel space)
                        <1>             ; esi = source address (virtual, in user's memory space)
00011CCC E863EEFFFF     <1>             call    transfer_from_user_buffer
00011CD1 720D           <1>             jc      short loc_swp_xor_retn
                        <1>
00011CD3 89E6           <1>             mov     esi, esp ; temporary buffer (the path) on stack
                        <1> loc_swp_fchar:
00011CD5 8A06           <1>             mov     al, [esi]
00011CD7 3C20           <1>             cmp     al, 20h
00011CD9 7711           <1>             ja      short loc_swp_parse_path_name
                        <1>             ;je     short loc_swp_fchar_next
                        <1>             ; 29/07/2022
00011CDB 7203           <1>             jb      short loc_swp_xor_retn
                        <1>
                        <1> loc_swp_fchar_next:
00011CDD 46             <1>             inc     esi
00011CDE EBF5           <1>             jmp     short loc_swp_fchar
                        <1>
                        <1> loc_swp_xor_retn:
00011CE0 31C0           <1>             xor     eax, eax
00011CE2 F9             <1>             stc
                        <1> loc_swp_retn:
00011CE3 89EC           <1>             mov     esp, ebp
00011CE5 5D             <1>             pop     ebp
                        <1>
                        <1>             ;mov    esi, FindFile_Drv
00011CE6 BE[24800100]   <1>             mov     esi, FindFile_Name ; 12/10/2016
00011CEB C3             <1>             retn
                        <1>
                        <1> ;loc_swp_fchar_next:
                        <1> ;           inc     esi
                        <1> ;           jmp     short loc_swp_fchar
                        <1>
                        <1> loc_swp_parse_path_name:
00011CEC BF[E27F0100]   <1>             mov     edi, FindFile_Drv
00011CF1 E8788CFFFF     <1>             call    parse_path_name
00011CF6 72EB           <1>             jc      short loc_swp_retn
                        <1>
                        <1> loc_swp_checkfile_name:
00011CF8 803D[3C830100]00 <1>           cmp     byte [SWP_Mode], 0
00011CFF 761E           <1>             jna     short loc_swp_drv
                        <1>
                        <1>             ; 10/10/2016 (valid file name checking)
00011D01 BE[24800100]   <1>             mov     esi, FindFile_Name
00011D06 803E20         <1>             cmp     byte [esi], 20h
00011D09 76D5           <1>             jna     short loc_swp_xor_retn
                        <1>
                        <1>             ; 16/10/2016
00011D0B C605[3B830100]00 <1>           mov     byte [SWP_inv_fname], 0 ; reset
                        <1>             ; esi = file name address (ASCIIZ)
00011D12 E8B26FFFFF     <1>             call    check_filename
00011D17 7306           <1>             jnc     short loc_swp_drv
                        <1>
00011D19 FE05[3B830100] <1>             inc     byte [SWP_inv_fname] ; set
                        <1> loc_swp_drv:
00011D1F 8A35[42770100] <1>             mov     dh, [Current_Drv]
                        <1>                ;mov       [RUN_CDRV], dh
                        <1>
00011D25 8A15[E27F0100] <1>             mov     dl, [FindFile_Drv]
                        <1>                ;cmp       dl, dh
00011D2B 3A15[42770100] <1>             cmp     dl, [Current_Drv]
00011D31 740D           <1>             je      short loc_swp_change_directory
```

```
1042                                <1>
1043 00011D33 FE05[3F830100]       <1>             inc     byte [SWP_DRV_chg]
1044 00011D39 E8A259FFFF           <1>             call    change_current_drive
1045 00011D3E 72A3                 <1>             jc      short loc_swp_retn ; eax = error code
1046                                <1>             ; eax = 0
1047                                <1>
1048                                <1> loc_swp_change_directory:
1049 00011D40 803D[E37F0100]21     <1>             cmp     byte [FindFile_Directory], 21h
1050 00011D47 F5                   <1>             cmc
1051 00011D48 7399                 <1>             jnc     short loc_swp_retn
1052                                <1>
1053 00011D4A FE05[3F830100]       <1>             inc     byte [SWP_DRV_chg]
1054 00011D50 FE05[5D2E0100]       <1>             inc     byte [Restore_CDIR]
1055 00011D56 BE[E37F0100]         <1>             mov     esi, FindFile_Directory
1056 00011D5B 8A25[3D830100]       <1>             mov     ah, [SWP_Mode+1]
1057 00011D61 E87586FFFF           <1>             call    change_current_directory
1058                                <1>             ;jc     short loc_swp_retn ; eax = error code
1059                                <1>             ; 08/08/2022
1060 00011D66 7305                 <1>             jnc     short loc_swp_change_prompt_dir_string
1061 00011D68 E976FFFFFF           <1>             jmp     loc_swp_retn
1062                                <1>
1063                                <1> loc_swp_change_prompt_dir_string:
1064                                <1>             ; esi = PATH_Array
1065                                <1>             ; eax = Current Directory First Cluster
1066                                <1>             ; edi = Logical DOS Drive Description Table
1067 00011D6D E89785FFFF           <1>             call    change_prompt_dir_str
1068 00011D72 29C0                 <1>             sub     eax, eax ; 0
1069 00011D74 E96AFFFFFF           <1>             jmp     loc_swp_retn
1070                                <1>
1071                                <1> reset_working_path:
1072                                <1>             ; 06/10/2016 - TRDOS 386 (TRDOS v2.0)
1073                                <1>             ;
1074                                <1>             ; TRDOS v1.0 (DIR.ASM, "proc_reset_working_path")
1075                                <1>             ; 05/02/2011 - 08/02/2011
1076                                <1>             ;
1077                                <1>             ; Restores current drive and directory
1078                                <1>             ;
1079                                <1>             ; INPUT: none
1080                                <1>             ; OUTPUT: DL = SWP_DRV, EAX = 0 -> OK
1081                                <1>             ;
1082                                <1>             ;    AX = 0 -> ESI = Logical Dos Drv Desc. Table
1083                                <1>             ;
1084                                <1>             ;    EAX, EBX, ECX, EDX, ESI, EDI will be changed
1085                                <1>             ;
1086                                <1>
1087                                <1>
1088 00011D79 31C0                 <1>             xor     eax, eax
1089 00011D7B 48                   <1>             dec     eax
1090                                <1>
1091 00011D7C 668B15[3E830100]     <1>             mov     dx, [SWP_DRV]
1092 00011D83 08F6                 <1>             or      dh, dh
1093 00011D85 742E                 <1>             jz      short loc_rwp_return
1094                                <1>
1095 00011D87 3A15[42770100]       <1>             cmp     dl, [Current_Drv]
1096 00011D8D 7407                 <1>             je      short loc_rwp_restore_cdir
1097                                <1> loc_rwp_restore_cdrv:
1098 00011D8F E84C59FFFF           <1>             call    change_current_drive
1099 00011D94 EB10                 <1>             jmp     short loc_rwp_restore_ok
1100                                <1> loc_rwp_restore_cdir:
1101 00011D96 31DB                 <1>             xor     ebx, ebx
1102 00011D98 88D7                 <1>             mov     bh, dl
1103 00011D9A BE00010900           <1>             mov     esi, Logical_DOSDisks
1104 00011D9F 01DE                 <1>             add     esi, ebx
1105                                <1>
1106 00011DA1 E8EB59FFFF           <1>             call    restore_current_directory
1107                                <1>
1108                                <1> loc_rwp_restore_ok:
1109 00011DA6 668B15[3E830100]     <1>             mov     dx, [SWP_DRV]
1110 00011DAD 31C0                 <1>             xor     eax, eax
1111 00011DAF 66A3[3F830100]       <1>             mov     [SWP_DRV_chg], ax
1112                                <1> loc_rwp_return:
1113 00011DB5 C3                   <1>             retn
1114                                <1>
1115                                <1> get_file_name:
1116                                <1>             ; 25/08/2024 (TRDOS 386 Kernel v2.0.9)
1117                                <1>             ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
1118                                <1>             ; 15/10/2016 - TRDOS 386 (TRDOS v2.0)
1119                                <1>             ; Convert file name
1120                                <1>             ;       from directory entry format
1121                                <1>                ;  to (8.3) dot file name format
1122                                <1>             ;
1123                                <1>             ; TRDOS v1.0 (DIR.ASM, "get_file_name")
1124                                <1>                ; 2005 - 09/10/2011
1125                                <1>             ; INPUT:
1126                                <1>             ;       DS:SI -> Directory Entry Format File Name
1127                                <1>             ;        ES:DI -> DOS Dot File Name Address
1128                                <1>             ; OUTPUT:
1129                                <1>             ;       DS:SI -> DOS Dot File Name Address
1130                                <1>                ;  ES:DI -> Directory Entry Format File Name
1131                                <1>             ;
1132                                <1>             ; TRDOS 386 (15/10/2016)
1133                                <1>             ; INPUT:
1134                                <1>             ;       ESI = File name addr in dir entry format
1135                                <1>             ;       EDI = Dot file name address (destination)
1136                                <1>             ; OUTPUT:
1137                                <1>             ;       File name is converted and moved
1138                                <1>             ;       to destination (as 8.3 dot filename)
1139                                <1>             ;
1140                                <1>             ; Modified registers: EAX, ECX
1141                                <1>
1142                                <1>                ; 2005 (TRDOS 8086) - 2016 (TRDOS 386)
1143                                <1>
1144 00011DB6 57                   <1>             push    edi
1145 00011DB7 56                   <1>             push    esi
1146 00011DB8 AC                   <1>             lodsb
1147                                <1>             ; 25/08/2024
1148 00011DB9 31C9                 <1>             xor     ecx, ecx ; 0
1149 00011DBB 3C20                 <1>             cmp     al, 20h
1150 00011DBD 7626                 <1>             jna     short pass_gfn_ext
1151                                <1>             ; 25/08/2024
1152                                <1>             ;push   esi
1153 00011DBF AA                   <1>             stosb
1154                                <1>             ; 25/08/2024
1155                                <1>             ; 29/07/2022
1156                                <1>             ;xor    ecx, ecx
1157                                <1>             ; ecx <= 128 ; 25/08/2024
1158 00011DC0 B107                 <1>             mov     cl, 7
1159                                <1>             ; 25/08/2024
1160 00011DC2 01CE                 <1>             add     esi, ecx ; add esi, 7
1161 00011DC4 56                   <1>             push    esi ; (*)
1162                                <1> loc_gfn_next_char:
1163 00011DC5 AC                   <1>             lodsb
1164 00011DC6 3C20                 <1>             cmp     al, 20h
1165 00011DC8 7603                 <1>             jna     short pass_gfn_fn
```

```
1166 00011DCA AA                <1>             stosb
1167 00011DCB E2F8              <1>             loop    loc_gfn_next_char
1168                            <1> pass_gfn_fn:
1169                            <1>             ;pop    esi
1170                            <1>             ;add    esi, 7
1171                            <1>             ; 25/08/2024
1172 00011DCD 5E                <1>             pop     esi ; (*)
1173                            <1>
1174 00011DCE AC                <1>             lodsb
1175 00011DCF 3C20              <1>             cmp     al, 20h
1176 00011DD1 7612              <1>             jna     short pass_gfn_ext
1177 00011DD3 B42E              <1>             mov     ah, '.'
1178 00011DD5 86C4              <1>             xchg    ah, al
1179 00011DD7 66AB              <1>             stosw
1180 00011DD9 AC                <1>             lodsb
1181 00011DDA 3C20              <1>             cmp     al, 20h
1182 00011DDC 7607              <1>             jna     short pass_gfn_ext
1183 00011DDE AA                <1>             stosb
1184 00011DDF AC                <1>             lodsb
1185 00011DE0 3C20              <1>             cmp     al, 20h
1186 00011DE2 7601              <1>             jna     short pass_gfn_ext
1187 00011DE4 AA                <1>             stosb
1188                            <1> pass_gfn_ext:
1189 00011DE5 30C0              <1>             xor     al, al
1190 00011DE7 AA                <1>             stosb
1191 00011DE8 5E                <1>             pop     esi
1192 00011DE9 5F                <1>             pop     edi
1193                            <1>             ; 25/08/2024
1194                            <1>             ; ecx <= 7
1195 00011DEA C3                <1>             retn
1196                            <1>
1197                            <1> set_hardware_int_vector:
1198                            <1>             ; 18/03/2017
1199                            <1>             ; 03/03/2017
1200                            <1>             ; 28/02/2017 - TRDOS 386 (TRDOS v2.0)
1201                            <1>             ;
1202                            <1>             ; SET/RESET HARDWARE INTERRUPT GATE
1203                            <1>             ;
1204                            <1>             ; Changes interrupt gate descriptor table
1205                            <1>             ; (without changing default interrupt list)
1206                            <1>             ;
1207                            <1>             ; INPUT:
1208                            <1>             ;       AL = IRQ number (0 to 15)
1209                            <1>             ;       AH > 0 -> set
1210                            <1>             ;       AH = 0 -> reset
1211                            <1>             ;
1212                            <1>             ; Modified registers: eax, ebx, edx, edi
1213                            <1>             ;
1214                            <1>
1215 00011DEB C0E002            <1>             shl     al, 2 ; IRQ number * 4
1216 00011DEE 0FB6D8            <1>             movzx   ebx, al
1217                            <1>
1218 00011DF1 08E4              <1>             or      ah, ah
1219 00011DF3 7508              <1>             jnz     short shintv_1 ; set (for user call service)
1220                            <1>
1221                            <1>             ; 18/03/2017
1222 00011DF5 81C3[48360100]    <1>             add     ebx, IRQ_list ; reset to default interrupt list
1223 00011DFB EB06              <1>             jmp     short shintv_2
1224                            <1> shintv_1:
1225 00011DFD 81C3[241E0100]    <1>             add     ebx, IRQ_u_list
1226                            <1> shintv_2:
1227 00011E03 8B13              <1>             mov     edx, [ebx] ; IRQ handler address
1228                            <1>
1229                            <1>             ; 03/03/2017
1230 00011E05 D0E0              <1>             shl     al, 1 ; IRQ number * 8
1231                            <1>             ; 18/03/2017
1232 00011E07 0FB6F8            <1>             movzx   edi, al
1233 00011E0A 81C7[98740100]    <1>             add     edi, idt + (8*32) ; IRQ 0 offset = idt + 256
1234                            <1>
1235 00011E10 89D0              <1>             mov     eax, edx ; IRQ handler address
1236 00011E12 BB00000800        <1>             mov     ebx, 80000h
1237                            <1>
1238                            <1>             ;mov    edx, eax
1239 00011E17 66BA008E          <1>             mov     dx, 8E00h
1240 00011E1B 6689C3            <1>             mov     bx, ax
1241 00011E1E 89D8              <1>             mov     eax, ebx ; /* selector = 0x0008 = cs */
1242                            <1>                                      ; /* interrupt gate - dpl=0, present */
1243 00011E20 AB                <1>             stosd   ; selector & offset bits 0-15
1244 00011E21 8917              <1>             mov     [edi], edx ; attributes & offset bits 16-23
1245                            <1>
1246 00011E23 C3                <1>             retn
1247                            <1> IRQ_u_list:
1248                            <1>             ; 28/02/2017
1249 00011E24 [88090000]        <1>             dd      timer_int
1250 00011E28 [ED100000]        <1>             dd      kb_int
1251 00011E2C [740B0000]        <1>             dd      irq2
1252 00011E30 [641E0100]        <1>             dd      IRQ_service3
1253 00011E34 [6E1E0100]        <1>             dd      IRQ_service4
1254 00011E38 [781E0100]        <1>             dd      IRQ_service5
1255 00011E3C [A04E0000]        <1>             dd      fdc_int
1256 00011E40 [821E0100]        <1>             dd      IRQ_service7
1257 00011E44 [FC0A0000]        <1>             dd      rtc_int
1258 00011E48 [8C1E0100]        <1>             dd      IRQ_service9
1259 00011E4C [961E0100]        <1>             dd      IRQ_service10
1260 00011E50 [A01E0100]        <1>             dd      IRQ_service11
1261 00011E54 [AA1E0100]        <1>             dd      IRQ_service12
1262 00011E58 [B41E0100]        <1>             dd      IRQ_service13
1263 00011E5C [DF570000]        <1>             dd      hdc1_int
1264 00011E60 [02580000]        <1>             dd      hdc2_int
1265                            <1>
1266                            <1>             ; 03/03/2017
1267                            <1>             ; 27/02/2017
1268                            <1> IRQ_service3:
1269 00011E64 36C605[AC880100]03 <1>             mov     byte [ss:IRQnum], 3
1270 00011E6C EB4E              <1>             jmp     short IRQ_service
1271                            <1> IRQ_service4:
1272 00011E6E 36C605[AC880100]04 <1>             mov     byte [ss:IRQnum], 4
1273 00011E76 EB44              <1>             jmp     short IRQ_service
1274                            <1> IRQ_service5:
1275 00011E78 36C605[AC880100]05 <1>             mov     byte [ss:IRQnum], 5
1276 00011E80 EB3A              <1>             jmp     short IRQ_service
1277                            <1> IRQ_service7:
1278 00011E82 36C605[AC880100]07 <1>             mov     byte [ss:IRQnum], 7
1279 00011E8A EB30              <1>             jmp     short IRQ_service
1280                            <1> IRQ_service9:
1281 00011E8C 36C605[AC880100]09 <1>             mov     byte [ss:IRQnum], 9
1282 00011E94 EB26              <1>             jmp     short IRQ_service
1283                            <1> IRQ_service10:
1284 00011E96 36C605[AC880100]0A <1>             mov     byte [ss:IRQnum], 10
1285 00011E9E EB1C              <1>             jmp     short IRQ_service
1286                            <1> IRQ_service11:
1287 00011EA0 36C605[AC880100]0B <1>             mov     byte [ss:IRQnum], 11
1288 00011EA8 EB12              <1>             jmp     short IRQ_service
1289                            <1> IRQ_service12:
```

```
1290 00011EAA 36C605[AC880100]0C  <1>             mov     byte [ss:IRQnum], 12
1291 00011EB2 EB08                <1>             jmp     short IRQ_service
1292                              <1> IRQ_service13:
1293 00011EB4 36C605[AC880100]0D  <1>             mov     byte [ss:IRQnum], 13
1294                              <1>             ;jmp    short IRQ_service
1295                              <1> IRQ_service:
1296                              <1>             ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
1297                              <1>             ; 13/06/2017
1298                              <1>             ; 11/06/2017
1299                              <1>             ; 10/06/2017
1300                              <1>             ; 01/03/2017, 04/03/2017
1301                              <1>             ; 27/02/2017, 28/02/2017
1302 00011EBC 1E                  <1>             push    ds
1303 00011EBD 06                  <1>             push    es
1304 00011EBE 0FA0                <1>             push    fs
1305 00011EC0 0FA8                <1>             push    gs
1306                              <1>
1307 00011EC2 60                  <1>             pushad  ; eax,ecx,edx,ebx,esp,ebp,esi,edi
1308 00011EC3 66B91000            <1>             mov     cx, KDATA
1309 00011EC7 8ED9                <1>             mov     ds, cx
1310 00011EC9 8EC1                <1>             mov     es, cx
1311 00011ECB 8EE1                <1>             mov     fs, cx
1312 00011ECD 8EE9                <1>             mov     gs, cx
1313                              <1>
1314 00011ECF 0F20D8              <1>             mov     eax, cr3
1315 00011ED2 A3[A8880100]        <1>             mov     [IRQ_cr3], eax
1316                              <1>
1317 00011ED7 A1[80760100]        <1>             mov     eax, [k_page_dir]
1318 00011EDC 0F22D8              <1>             mov     cr3, eax
1319                              <1>
1320 00011EDF A0[AC880100]        <1>             mov     al, [IRQnum]
1321                              <1>
1322                              <1>             ;mov    cl, [sysflg]
1323                              <1>             ;mov    [u.r_mode], cl  ; system (0) or user mode (FFh)
1324                              <1> IRQsrv_0:
1325 00011EE4 0FB6D8              <1>             movzx   ebx, al
1326 00011EE7 8A9B[80350100]      <1>             mov     bl, [ebx+IRQenum] ; IRQ (available) index number + 1
1327                              <1>             ; 01/03/2017
1328 00011EED FECB                <1>             dec     bl  ; IRQ index number, 0 to 8
1329                              <1>             ;js     IRQsrv_5 ; not available to use here!?
1330                              <1>             ; 29/07/2022
1331 00011EEF 785E                <1>             js      short IRQsrv_j5  ; (jump to IRQsrv_5)
1332                              <1>             ;
1333 00011EF1 80BB[72880100]80    <1>             cmp     byte [ebx+IRQ.method], 80h ; using by a dev or kernel?
1334 00011EF8 7205                <1>             jb      short IRQsrv_1 ; no
1335                              <1>
1336                              <1>             ; If the IRQ service is already owned by TRDOS 386 kernel
1337                              <1>             ;       or a Device driver
1338                              <1>             ; we need to call 'dev_IRQ_service'
1339                              <1>
1340                              <1>             ; IRQ number in AL
1341 00011EFA E80E010000            <1>           call    dev_IRQ_service         ; IRQ service for device drivers
1342                              <1>             ; IRQ number in AL
1343                              <1> IRQsrv_1:
1344                              <1>             ; check user callback service status
1345                              <1>             ; AL = IRQ number
1346                              <1>             ; EBX = IRQ (Available) Index number
1347                              <1>
1348 00011EFF A2[AF8E0100]        <1>             mov     [u.irqwait], al ; set waiting IRQ flag
1349                              <1>
1350 00011F04 8A83[60880100]      <1>             mov     al, [ebx+IRQ.owner]
1351 00011F0A 20C0                <1>             and     al, al
1352                              <1>             ;jz     IRQsrv_5 ; it is not owned by a user/proc
1353                              <1>             ; 29/07/2022
1354 00011F0C 7441                <1>             jz      short IRQsrv_j5  ; (jump to IRQsrv_5)
1355                              <1>
1356                              <1>             ; 03/03/2017
1357 00011F0E 89DA                <1>             mov     edx, ebx
1358 00011F10 C0E202              <1>             shl     dl, 2
1359 00011F13 8B92[84880100]      <1>             mov     edx, [edx+IRQ.addr] ; S.R.B. or Callback service addr
1360                              <1>
1361 00011F19 8AA3[72880100]      <1>             mov     ah, [ebx+IRQ.method]
1362 00011F1F F6C401              <1>             test    ah, 1
1363 00011F22 7530                <1>             jnz     short IRQsrv_4 ; Callback service method
1364                              <1>
1365                              <1>             ; Signal Response Byte method
1366                              <1>             ;mov    edx, [edx+IRQ.addr] ; Signal Response Byte address
1367                              <1>             ;                           ; (Physical address, non-swappable)
1368 00011F24 80E402              <1>             and     ah, 2 ; bit 1, (S.R.B.) counter (auto increment) method
1369 00011F27 8AA3[7B880100]      <1>             mov     ah, [ebx+IRQ.srb] ; Signal Response Byte value
1370 00011F2D 7408                <1>             jz      short IRQsrv_2 ; fixed S.R.B. value
1371                              <1>             ; counter method (auto increment)
1372 00011F2F FEC4                <1>             inc     ah
1373 00011F31 88A3[7B880100]      <1>             mov     [ebx+IRQ.srb], ah ; next (count) number
1374                              <1> IRQsrv_2:
1375 00011F37 8822                <1>             mov     [edx], ah ; put S.R.B. val to the user's S.R.B. addr
1376 00011F39 C605[AF8E0100]00    <1>             mov     byte [u.irqwait], 0 ; clear waiting IRQ flag
1377                              <1>
1378 00011F40 3A05[858E0100]      <1>             cmp     al, [u.uno]
1379                              <1>             ;je     IRQsrv_5 ; the owner is current user/process
1380                              <1>             ; 29/07/2022
1381 00011F46 7407                <1>             je      short IRQsrv_j5  ; (jump to IRQsrv_5)
1382                              <1> IRQsrv_3:
1383                              <1>             ; the owner is not current user/process
1384                              <1>             ; AL = process number
1385 00011F48 B202                <1>             mov     dl, 2 ; priority, 2 = event (high)
1386 00011F4A E87BFBFFFF          <1>             call    set_run_sequence
1387                              <1>
1388                              <1>             ; [u.irqwait] = waiting IRQ number for callback service
1389                              <1> IRQsrv_j5:          ; 29/07/2022
1390 00011F4F E998000000          <1>             jmp     IRQsrv_5
1391                              <1> IRQsrv_4:
1392 00011F54 3A05[858E0100]      <1>             cmp     al, [u.uno]  ; is the owner is current user/process?
1393 00011F5A 75EC                <1>             jne     short IRQsrv_3 ; no !
1394                              <1>
1395                              <1>             ; Check if an IRQ callback service already in progress
1396 00011F5C 803D[B08E0100]00    <1>             cmp     byte [u.r_lock], 0
1397                              <1>             ;ja     IRQsrv_5 ; nothing to do !
1398                              <1>                             ; (we need to complete prev callback)
1399                              <1>             ; 29/07/2022
1400 00011F63 77EA                <1>             ja      short IRQsrv_j5  ; (jump to IRQsrv_5)
1401                              <1>
1402 00011F65 803D[AC8E0100]00    <1>             cmp     byte [u.t_lock], 0
1403 00011F6C 777E                <1>             ja      short IRQsrv_5 ; nothing to do !
1404                              <1>                             ; (we need to complete timer callback)
1405                              <1>
1406                              <1>             ; 04/03/2017
1407 00011F6E C605[AF8E0100]00    <1>             mov     byte [u.irqwait], 0 ; reset/clear waiting IRQ flag
1408                              <1>
1409 00011F75 FE05[B08E0100]      <1>             inc     byte [u.r_lock] ; 'IRQ callback service in progress' flag
1410                              <1>
1411 00011F7B 8A0D[288E0100]      <1>             mov     cl, [sysflg]   ; (system call) mode flag (kernel/user)
1412 00011F81 880D[B18E0100]      <1>             mov     [u.r_mode], cl ; system mode (0) or user mode (FFh)
1413                              <1>
```

```
1414                                    <1>             ;
1415 00011F87 8B2D[1C760100]           <1>             mov     ebp, [tss.esp0] ; kernel stack address (for ring 0)
1416 00011F8D 83ED14                    <1>             sub     ebp, 20         ; eip, cs, eflags, esp, ss
1417 00011F90 892D[2C8E0100]            <1>             mov     [u.sp], ebp
1418 00011F96 8925[308E0100]            <1>             mov     [u.usp], esp
1419                                    <1>
1420                                    <1>             ;or     word [ebp+8], 200h ; 22/01/2017, force enabling interrupts
1421                                    <1>
1422 00011F9C 8B44241C                  <1>             mov     eax, [esp+28] ; pushed eax
1423 00011FA0 A3[348E0100]              <1>             mov     [u.r0], eax
1424                                    <1>
1425 00011FA5 E8FCE9FFFF                <1>             call    wswap ; save user's registers & status
1426                                    <1>
1427                                    <1>             ; software int is in ring 0 but IRQ handler must return to ring 3
1428                                    <1>             ; so, ring 3 return address and stack registers
1429                                    <1>             ; (eip, cs, eflags, esp, ss)
1430                                    <1>             ; must be copied to IRQ handler return
1431                                    <1>             ; eip will be replaced by callback service routine address
1432                                    <1>
1433 00011FAA C605[288E0100]FF          <1>             mov     byte [sysflg], 0FFh ; user mode
1434                                    <1>
1435                                    <1>             ; system mode (system call)
1436                                    <1>             ;mov    ebp, [u.sp] ; EIP (u), CS (UCODE), EFLAGS (u),
1437                                    <1>                                 ; ESP (u), SS (UDATA)
1438                                    <1>
1439 00011FB1 8B4510                    <1>             mov     eax, [ebp+16]  ; SS (UDATA)
1440 00011FB4 89E6                      <1>             mov     esi, esp
1441 00011FB6 50                        <1>             push    eax
1442 00011FB7 50                        <1>             push    eax
1443 00011FB8 89E7                      <1>             mov     edi, esp
1444 00011FBA 893D[308E0100]            <1>             mov     [u.usp], edi
1445 00011FC0 B908000000                <1>             mov     ecx, ((ESPACE/4) - 4) ; except DS, ES, FS, GS
1446 00011FC5 F3A5                      <1>             rep     movsd
1447 00011FC7 B104                      <1>             mov     cl, 4
1448 00011FC9 F3AB                      <1>             rep     stosd
1449 00011FCB 893D[2C8E0100]            <1>             mov     [u.sp], edi
1450 00011FD1 89EE                      <1>             mov     esi, ebp
1451 00011FD3 B105                      <1>             mov     cl, 5 ; EIP (u), CS (UCODE), EFLAGS (u), ESP (u), SS (UDATA)
1452 00011FD5 F3A5                      <1>             rep     movsd
1453                                    <1>             ;
1454                                    <1>
1455 00011FD7 8B0D[8C8E0100]            <1>             mov     ecx, [u.pgdir]
1456 00011FDD 890D[A8880100]            <1>             mov     [IRQ_cr3], ecx
1457                                    <1>
1458                                    <1> set_IRQ_callback_addr:
1459                                    <1>             ;
1460                                    <1>             ; This routine sets return address
1461                                    <1>             ; to start of user's interrupt
1462                                    <1>             ; service (callback) address
1463                                    <1>             ;
1464                                    <1>             ; INPUT:
1465                                    <1>             ;       EDX = callback routine/service address
1466                                    <1>             ;             (virtual, not physical address!)
1467                                    <1>             ;       [u.sp] = kernel stack, points to
1468                                    <1>             ;                user's EIP,CS,EFLAGS,ESP,SS
1469                                    <1>             ;                registers.
1470                                    <1>             ; OUTPUT:
1471                                    <1>             ;       EIP (user) = callback (service) address
1472                                    <1>             ;       CS (user) = UCODE
1473                                    <1>             ;       EFLAGS (user) = flags before callback
1474                                    <1>             ;        ESP (user) = ESP-4 (user, before callback)
1475                                    <1>             ;       [ESP](user) = EIP (user) before callback
1476                                    <1>             ;
1477                                    <1>             ; Note: If CPU was in user mode while entering
1478                                    <1>             ;       the timer interrupt service routine,
1479                                    <1>             ;       'IRET' will get return to callback routine
1480                                    <1>             ;       immediately. If CPU was in system/kernel mode
1481                                    <1>             ;       'iret' will get return to system call and
1482                                    <1>             ;       then, callback routine will be return address
1483                                    <1>             ;       from system call. (User's callback/service code
1484                                    <1>             ;       will be able to return to normal return address
1485                                    <1>             ;       via a 'sysrele' system call at the end.)
1486                                    <1>             ;
1487                                    <1>             ; Note: User's IRQ callback service code must be ended
1488                                    <1>             ;       with a 'sysrele' system call !
1489                                    <1>             ;
1490                                    <1>             ;       For example:
1491                                    <1>             ;
1492                                    <1>             ;       audio_IRQ_callback:
1493                                    <1>             ;           ...
1494                                    <1>             ;           <load DMA buffer with audio data>
1495                                    <1>             ;           ...
1496                                    <1>             ;           mov eax, 39 ; 'sysrele'
1497                                    <1>             ;           int 40h ; TRDOS 386 system call (interrupt)
1498                                    <1>             ;
1499                                    <1>
1500                                    <1>             ;mov    edx, [edx+IRQ.addr] ; Callback service address
1501                                    <1>             ;                           ; (Virtual address)
1502                                    <1>
1503 00011FE3 8B2D[2C8E0100]            <1>             mov     ebp, [u.sp]; kernel's stack, points to EIP (user)
1504 00011FE9 895500                    <1>             mov     [ebp], edx
1505                                    <1> IRQsrv_5:
1506                                    <1>             ; EOI & return
1507                                    <1>             ; 01/08/2020
1508                                    <1>             ; 11/06/2017
1509                                    <1>             ; 10/06/2017
1510                                    <1>             ;mov    al, [IRQnum]
1511 00011FEC B020                      <1>             mov     al, 20h ; 01/08/2020
1512 00011FEE FA                        <1>             cli
1513                                    <1>             ;cmp    al, 7
1514 00011FEF 803D[AC880100]07          <1>             cmp     byte [IRQnum], 7 ; 01/08/2020
1515 00011FF6 7602                      <1>             jna     short IRQsrv_6
1516                                    <1>             ;
1517                                    <1>             ;;mov   al, EOI ; end of interrupt
1518                                    <1>             ;mov    al, 20h ; 01/08/2020
1519                                    <1>             ;cli            ; disable interrupts till stack cleared
1520                                    <1>             ;out    INTB00, al ; For controll2 #2
1521 00011FF8 E6A0                      <1>             out     0A0h, al
1522                                    <1> IRQsrv_6:
1523                                    <1>             ;mov    byte [IRQnum], 0 ; reset
1524                                    <1>             ;;mov   al, EOI ; end of interrupt
1525                                    <1>             ;mov    al, 20h ; 01/08/2020
1526                                    <1>             ;cli            ; disable interrupts till stack cleared
1527                                    <1>             ;out    INTA00, al ; end of interrupt to 8259 - 1
1528 00011FFA E620                      <1>             out     20h, al
1529                                    <1> IRQsrv_7:
1530                                    <1>             ;; 13/06/2017
1531                                    <1>             ;or     word [ebp+8], 200h ; force enabling interrupts
1532                                    <1>             ;
1533 00011FFC 8B0D[A8880100]            <1>             mov     ecx, [IRQ_cr3] ; previous content of cr3 register
1534 00012002 0F22D9                    <1>             mov     cr3, ecx       ; restore cr3 register content
1535                                    <1>             ;
1536 00012005 61                        <1>             popad ; edi,esi,ebp,(icrement esp by 4),ebx,edx,ecx,eax
1537                                    <1>             ;
```

```
1538 00012006 0FA9              <1>             pop     gs
1539 00012008 0FA1              <1>             pop     fs
1540 0001200A 07                <1>             pop     es
1541 0001200B 1F                <1>             pop     ds
1542                            <1>             ;
1543 0001200C CF                <1>             iretd   ; return from interrupt
1544                            <1>
1545                            <1> ; 17/04/2021
1546                            <1> ; ('get_device_number' procedure is disabled as temporary)
1547                            <1>
1548                            <1> ;get_device_number:
1549                            <1> ;           ; 08/10/2016
1550                            <1> ;           ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
1551                            <1> ;           ;
1552                            <1> ;           ; This procedure compares name of requested
1553                            <1> ;           ; device with kernel device names and
1554                            <1> ;           ; installable device names. If names match,
1555                            <1> ;           ; the relevant device index (entry) number
1556                            <1> ;           ; will be returned the caller (sysopen)
1557                            <1> ;           ; for the requested device.
1558                            <1> ;           ;
1559                            <1> ;           ; NOTE: Installable device drivers must
1560                            <1> ;           ; be loaded before using 'sysopen'
1561                            <1> ;           ; (opendev) system call.
1562                            <1> ;           ;
1563                            <1> ;           ; INPUT:
1564                            <1> ;           ;    ESI = device name address (ASCIIZ)
1565                            <1> ;           ;         (in kernel's memory space)
1566                            <1> ;           ;    max name length = 8 without '/dev/')
1567                            <1> ;           ;    Device name will be capitalized
1568                            <1> ;           ;    and if there is, '/dev/' will be
1569                            <1> ;           ;    removed from name before comparising)
1570                            <1> ;           ;
1571                            <1> ;           ; OUTPUT:
1572                            <1> ;           ;    cf = 0 ->
1573                            <1> ;           ;      EAX (AL) = device entry/index number
1574                            <1> ;           ;    cf = 1 -> device not found (installed)
1575                            <1> ;           ;              or invalid device name
1576                            <1> ;           ;              (AL=0)
1577                            <1> ;           ;    device_name = device name address (asciiz)
1578                            <1> ;           ;          ;
1579                            <1> ;           ; Modified registers: EAX, EBX, ESI, EDI
1580                            <1> ;
1581                            <1> ;           mov     edi, device_name
1582                            <1> ;           call    lodsb_capitalize
1583                            <1> ;           mov     ah, al
1584                            <1> ;           cmp     al, '/'
1585                            <1> ;           jne     short gdn_1
1586                            <1> ;           mov     edi, device_name
1587                            <1> ;           call    lodsb_capitalize
1588                            <1> ;gdn_0:
1589                            <1> ;           and     al, al ; 0 ?
1590                            <1> ;           jz      short gdn_err ; null name after '/'
1591                            <1> ;gdn_1:
1592                            <1> ;           cmp     al, 'D'
1593                            <1> ;           jne     short gdn_2
1594                            <1> ;           call    lodsb_capitalize
1595                            <1> ;           cmp     al, 'E'
1596                            <1> ;           jne     short gdn_2
1597                            <1> ;           call    lodsb_capitalize
1598                            <1> ;           cmp     al, 'V'
1599                            <1> ;           jne     short gdn_2
1600                            <1> ;           lodsb
1601                            <1> ;           cmp     al, '/'
1602                            <1> ;           je      short gdn_4
1603                            <1> ;gdn_2:
1604                            <1> ;           cmp     ah, '/'
1605                            <1> ;           jne     short gdn_5
1606                            <1> ;gdn_err:
1607                            <1> ;           ; invalid device name or device not found
1608                            <1> ;           xor     eax, eax ; 0
1609                            <1> ;           stc
1610                            <1> ;           retn
1611                            <1> ;gdn_3:
1612                            <1> ;           cmp     al, '/'
1613                            <1> ;           jne     short gdn_5
1614                            <1> ;gdn_4:
1615                            <1> ;           mov     edi, device_name
1616                            <1> ;           jmp     short gdn_6
1617                            <1> ;gdn_5:
1618                            <1> ;           cmp     al, 0
1619                            <1> ;           je      short gdn_7
1620                            <1> ;gdn_6:
1621                            <1> ;           call    lodsb_capitalize
1622                            <1> ;           cmp     edi, device_name + 8
1623                            <1> ;           jb      short gdn_3
1624                            <1> ;           cmp     al, 0
1625                            <1> ;           jne     short gdn_err
1626                            <1> ;           cmp     edi, device_name + 1
1627                            <1> ;           jna     short gdn_err ; null name after '/'
1628                            <1> ;gdn_7:
1629                            <1> ;           stosb
1630                            <1> ;           ; zero padding ("NAME",0,0,0,0)
1631                            <1> ;           cmp     edi, device_name + 8
1632                            <1> ;           jb      short gdn_7
1633                            <1> ;gdn_8:
1634                            <1> ;           ; search for kernel device names
1635                            <1> ;           mov     esi, device_name
1636                            <1> ;           mov     edi, KDEV_NAME
1637                            <1> ;           xor     eax, eax
1638                            <1> ;gdn_9:
1639                            <1> ;           cmpsd
1640                            <1> ;           jne     short gdn_10
1641                            <1> ;           cmpsd
1642                            <1> ;           jne     short gdn_11
1643                            <1> ;           jmp     short gdn_17 ; match
1644                            <1> ;gdn_10:
1645                            <1> ;           cmpsd  ; add esi, 4 & add edi, 4
1646                            <1> ;gdn_11:
1647                            <1> ;           mov     esi, device_name
1648                            <1> ;           inc     al
1649                            <1> ;           cmp     al, NumOfKernelDevNames
1650                            <1> ;           jb      short gdn_9
1651                            <1> ;gdn_12:
1652                            <1> ;           ; search for installable device names
1653                            <1> ;           ; esi = offset device_name
1654                            <1> ;           mov     edi, IDEV_NAME
1655                            <1> ;           sub     al, al ; 0
1656                            <1> ;gdn_13:
1657                            <1> ;           cmpsd
1658                            <1> ;           jne     short gdn_14
1659                            <1> ;           cmpsd
1660                            <1> ;           jne     short gdn_15
1661                            <1> ;           jmp     short gdn_19 ; match
```

```
1662                    <1> ;gdn_14:
1663                    <1> ;           cmpsd  ; add esi, 4 & add edi, 4
1664                    <1> ;gdn_15:
1665                    <1> ;           mov    esi, device_name
1666                    <1> ;           inc    al
1667                    <1> ;           cmp    al, NumOfInstallableDevices
1668                    <1> ;           jb     short gdn_13
1669                    <1> ;
1670                    <1> ;gdn_16:    ; error: invalid device name (not found) !
1671                    <1> ;           xor    al, al
1672                    <1> ;           stc
1673                    <1> ;           retn
1674                    <1> ;
1675                    <1> ;gdn_17:    ; name match (with one of kernel device names)
1676                    <1> ;           ;
1677                    <1> ;           ; convert KDEV_NAME index to
1678                    <1> ;           ; KDEV_NUMBER index
1679                    <1> ;           ; (different names are used for same devices)
1680                    <1> ;           ; (example: "COM1" & "TTY8" = device number 18)
1681                    <1> ;           mov    ebx, eax ; < 256
1682                    <1> ;           mov    al, [KDEV_NUMBER+ebx]
1683                    <1> ;
1684                    <1> ;           ; check if empty dev entry in the list
1685                    <1> ;           cmp    byte [DEV_OPENMODE+eax], 0
1686                    <1> ;           ja     short gdn_18 ; it must be already set
1687                    <1> ;
1688                    <1> ;           ; (re)set device name and access flags
1689                    <1> ;           ; (remain open work will be easy after that)
1690                    <1> ;           ; (NOTE: here, data will be copied to bss section)
1691                    <1> ;           mov    bl, al
1692                    <1> ;           sub    edi, 8 ; kernel device name address (data)
1693                    <1> ;           shl    bx, 2
1694                    <1> ;           mov    [DEV_NAME_PTR+ebx], edi ; (all) device names
1695                    <1> ;           mov    bl, [KDEV_ACCESS+eax] ; kernel dev list (data)
1696                    <1> ;           mov    [DEV_ACCESS+eax], bl ; (all) device list (bss)
1697                    <1> ;gdn_18:
1698                    <1> ;           inc    al ; 1 to NumOfKernelDevNames (<=7Fh)
1699                    <1> ;           ; eax = device index/entry number
1700                    <1> ;           retn
1701                    <1> ;
1702                    <1> ;gdn_19:    ; name match (with one of installable device names)
1703                    <1> ;           ;
1704                    <1> ;           ; al = 0 to NumOfInstallableDevices - 1 (<=7Fh)
1705                    <1> ;
1706                    <1> ;           mov    ebx, eax
1707                    <1> ;           add    bl, NumOfKernelDevices       ; < NUMOFDEVICES
1708                    <1> ;
1709                    <1> ;           ; check if empty dev entry in the list
1710                    <1> ;           cmp    byte [DEV_OPENMODE+ebx], 0
1711                    <1> ;           ja     short gdn_20 ; it must be already set
1712                    <1> ;
1713                    <1> ;           ; (re)set device name and access flags
1714                    <1> ;           ; (remain open work will be easy after that)
1715                    <1> ;           sub    edi, 8 ; installable device name address
1716                    <1> ;           shl    bx, 2 ;*4
1717                    <1> ;           mov    [DEV_NAME_PTR+ebx], edi ; (all) device names
1718                    <1> ;           shr    bx, 2
1719                    <1> ;           mov    al, [IDEV_FLAGS+eax] ; installable dev list
1720                    <1> ;           mov    [DEV_ACCESS+ebx], al ; (all) device list
1721                    <1> ;gdn_20:
1722                    <1> ;           mov    al, bl
1723                    <1> ;           ; eax = device index/entry number ; < NUMOFDEVICES
1724                    <1> ;           retn
1725                    <1>
1726                    <1> ;lodsb_capitalize:
1727                    <1> ;   ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
1728                    <1> ;   ; INPUT -> [esi] = character
1729                    <1> ;   ;          edi = destination
1730                    <1> ;   ; OUTPUT -> AL contains capitalized character
1731                    <1> ;   ;          esi = esi+1
1732                    <1> ;   ;          edi = edi+1
1733                    <1> ;   ;
1734                    <1> ;   lodsb
1735                    <1> ;   cmp    al, 61h
1736                    <1> ;           jb     short lodsb_cap_retn
1737                    <1> ;           cmp    al, 7Ah
1738                    <1> ;           ja     short lodsb_cap_retn
1739                    <1> ;           and    al, 0DFh
1740                    <1> ;lodsb_cap_retn:
1741                    <1> ;   stosb
1742                    <1> ;   retn
1743                    <1>
1744                    <1> ; 17/04/2021
1745                    <1> ; ('device_open' procedure is disabled as temporary)
1746                    <1>
1747                    <1> ;device_open:
1748                    <1> ;   ; 08/10/2016 - TRDOS 386 (TRDOS v2.0)
1749                    <1> ;   ; Complete device opening work for sysopen (device)
1750                    <1> ;   ;
1751                    <1> ;   ; INPUT ->
1752                    <1> ;   ;       EAX = Device Number (AL)
1753                    <1> ;   ;         CL = Open mode (1 = read, 2 = write)
1754                    <1> ;   ;        CH = Device access byte (bit 0 = 0)
1755                    <1> ;   ; OUTPUT ->
1756                    <1> ;   ;       EAX = Device Number
1757                    <1> ;   ;       CF = 0 -> device has been opened
1758                    <1> ;   ;       CF = 1 -> device could not be opened
1759                    <1> ;   ;
1760                    <1> ;   ;  Modified registers: ebx, (edx, ecx, esi, edi, ebp)
1761                    <1> ;   ;
1762                    <1> ;
1763                    <1> ;   mov    ebx, eax
1764                    <1> ;   shl    bx, 2 ; *4
1765                    <1> ;
1766                    <1> ;   test   ch, 80h ; bit 7, installable device driver flag
1767                    <1> ;   jz     short d_open_2 ; Kernel device
1768                    <1> ;   ; installable device
1769                    <1> ;d_open_1:
1770                    <1> ;       jmp dword [ebx+IDEV_OADDR-4]
1771                    <1> ;d_open_2:
1772                    <1> ;   jmp    dword [ebx+KDEV_OADDR-4]
1773                    <1>
1774                    <1> ; 17/04/2021
1775                    <1> ; ('device_close' procedure is disabled as temporary)
1776                    <1>
1777                    <1> ;device_close:
1778                    <1> ;   ; 08/10/2016 - TRDOS 386 (TRDOS v2.0)
1779                    <1> ;   ; Complete device closing work for sysclose (device)
1780                    <1> ;   ;
1781                    <1> ;   ; INPUT ->
1782                    <1> ;   ;       EAX = Device Number (AL)
1783                    <1> ;   ;         CL = Open mode (1 = read, 2 = write)
1784                    <1> ;   ;        CH = Device access byte (bit 0 = 0)
1785                    <1> ;   ; OUTPUT ->
```

```
                        <1> ;   ;       EAX = Device Number
                        <1> ;   ;       CF = 0 -> device has been closed
                        <1> ;   ;       CF = 1 -> device could not be closed
                        <1> ;   ;
                        <1> ;   ; Modified registers: ebx, (edx, ecx, esi, edi, ebp)
                        <1> ;   ;
                        <1> ;
                        <1> ;   mov     ebx, eax
                        <1> ;   shl     bx, 2 ; *4
                        <1> ;
                        <1> ;   test    ch, 80h ; bit 7, installable device driver flag
                        <1> ;   jz      short d_close_2 ; Kernel device
                        <1> ;   ; installable device
                        <1> ;d_close_1:
                        <1> ;       jmp dword [ebx+IDEV_CADDR-4]
                        <1> ;d_close_2:
                        <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)
                        <1> ;   retn
                        <1>
                        <1> ;wnull:
                        <1> ;   ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
                        <1> ;   ; write null (write to null device)
                        <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>     ;
0001200D 53             <1>     push    ebx
0001200E 0FB6D8         <1>     movzx   ebx, al
00012011 C0E302         <1>     shl     bl, 2 ; * 4
00012014 8B9B[20880100] <1>     mov     ebx, [ebx+DEV_INT_HNDLR]
0001201A 21DB           <1>     and     ebx, ebx
0001201C 7404           <1>         jz  short dIRQ_s_retn
0001201E 50             <1>     push    eax
                        <1>
0001201F FFD3           <1>     call    ebx
                        <1>
00012021 58             <1>     pop     eax
                        <1> dIRQ_s_retn:
00012022 5B             <1>     pop     ebx
00012023 C3             <1>     retn
                        <1>
                        <1> set_dev_IRQ_service:
                        <1>     ; 09/08/2022
                        <1>     ; 29/07/2022 (TRDOS 386 Kernel v2.0.5)
                        <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
00012024 0FB6F0         <1>     movzx   esi, al
                        <1>     ;shl    si, 2 ; * 4
                        <1>     ; 09/08/2022
00012027 C1E602         <1>     shl     esi, 2 ; * 4
0001202A 899E[20880100] <1>     mov     [esi+DEV_INT_HNDLR], ebx
                        <1>     ;pop    esi
00012030 C3             <1>     retn
                        <1>
                        <1> sysaudio: ; AUDIO FUNCTIONS
                        <1>     ; 28/01/2025
                        <1>     ; 27/01/2025
                        <1>     ; 15/01/2025
                        <1>     ; 12/01/2025
                        <1>     ; 11/01/2025 (TRDOS 386 v2.0.10)
                        <1>     ; 29/12/2024
                        <1>     ; 19/12/2024
                        <1>     ; 23/08/2024 (TRDOS 386 v2.0.9)
                        <1>     ; 05/06/2024
                        <1>     ; 04/06/2024
                        <1>     ; 23/05/2024 (TRDOS 386 v2.0.8)
                        <1>     ; 19/11/2023 (TRDOS 386 v2.0.7)
                        <1>     ; 29/07/2022 (TRDOS 386 v2.0.5)
                        <1>     ; 12/02/2021 (TRDOS 386 v2.0.3)
                        <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>     ;
```

```
1910                    <1>     ;       BH = 2 -> ALLOCATE AUDIO BUFFER (for user)
1911                    <1>     ;               ECX = Audio Buffer Size (must be equal to
1912                    <1>     ;                     the half of DMA buffer size)
1913                    <1>     ;               EDX = Virtual Address of the buffer
1914                    <1>     ;                     (This is not DMA buffer!)
1915                    <1>     ;
1916                    <1>     ;       BH = 3 -> INITIALIZE AUDIO DEVICE
1917                    <1>     ;            BL = 0,2 -> for Signal Response Byte
1918                    <1>     ;              CL = Signal Response Byte Value (fixed)
1919                    <1>     ;                                   if BL = 0
1920                    <1>     ;                           auto increment of S.R.B. value
1921                    <1>     ;                                   if BL = 2
1922                    <1>     ;                 EDX = Signal Response (Return) Byte Address
1923                    <1>     ;
1924                    <1>     ;            BL = 1 for CallBack Method
1925                    <1>     ;              EDX = CallBack Service Address (Virtual)
1926                    <1>     ;
1927                    <1>     ;            BL > 2 -> invalid function
1928                    <1>     ;
1929                    <1>     ;           (Audio buffer must be allocated before
1930                    <1>     ;            initialization.)
1931                    <1>     ;
1932                    <1>     ;       BH = 4 -> START TO PLAY
1933                    <1>     ;            BL = Mode
1934                    <1>     ;                 Bit 0 = mono/stereo (1 = stereo)
1935                    <1>     ;                 Bit 1 = 8 bit / 16 bit (1 = 16 bit)
1936                    <1>     ;            CX = Sampling Rate (Hz)
1937                    <1>     ;
1938                    <1>     ;       BH = 5 -> PAUSE
1939                    <1>     ;            BL = Any
1940                    <1>     ;
1941                    <1>     ;       BH = 6 -> CONTINUE TO PLAY
1942                    <1>     ;            BL = Any
1943                    <1>     ;
1944                    <1>     ;       BH = 7 -> STOP
1945                    <1>     ;            BL = Any
1946                    <1>     ;
1947                    <1>     ;       BH = 8 -> RESET
1948                    <1>     ;            BL = Any
1949                    <1>     ;
1950                    <1>     ;       BH = 9 -> CANCEL (CALLBACK or S.R.B. SERVICE)
1951                    <1>     ;            BL = Any
1952                    <1>     ;
1953                    <1>     ;       BH = 10 -> DEALLOCATE AUDIO BUFFER (for user)
1954                    <1>     ;            BL = Any
1955                    <1>     ;
1956                    <1>     ;       BH = 11 -> SET VOLUME LEVEL
1957                    <1>     ;            BL: (Bit 0 to 6)
1958                    <1>     ;                0 = Master (Playback, Lineout) volume
1959                    <1>     ;                1 = PCM out volume ; 23/05/2024
1960                    <1>     ;            CL = Left Channel Volume (0 to 31 max)
1961                    <1>     ;            CH = Right Channel Volume (0 to 31 max)
1962                    <1>     ;
1963                    <1>     ;            Note: If BL >= 80h (Bit 7 of BL is set),
1964                    <1>     ;            volume level will be set for next playing
1965                    <1>     ;            (actual volume level will not be changed
1966                    <1>     ;            immediately)
1967                    <1>     ;
1968                    <1>     ;       BH = 12 -> DISABLE AUDIO DEVICE
1969                    <1>     ;            (reset audio device and unlink dma buffer)
1970                    <1>     ;            BL = Any
1971                    <1>     ;
1972                    <1>     ;       12/05/2017
1973                    <1>     ;       BH = 13 -> MAP DMA BUFFER TO USER
1974                    <1>     ;           (for direct access to system's dma buffer)
1975                    <1>     ;
1976                    <1>     ;            ECX = map size in bytes
1977                    <1>     ;                (will be rounded up to page borders)
1978                    <1>     ;            EDX = Virtual Address of the buffer
1979                    <1>     ;                (Will be rounded up to page borders)
1980                    <1>     ;
1981                    <1>     ;       05/06/2017
1982                    <1>     ;       04/06/2017
1983                    <1>     ;       BH = 14 -> GET AUDIO DEVICE INFO
1984                    <1>     ;            BL: 0 = Audio Controller Info
1985                    <1>     ;            ; 19/11/2023
1986                    <1>     ;            BL: 1 = Audio (AC'97) Codec Info
1987                    <1>     ;            BL > 1 = Invalid for now!
1988                    <1>     ;
1989                    <1>     ;       22/06/2017
1990                    <1>     ;       BH = 15 -> GET CURRENT SOUND DATA (for graphics)
1991                    <1>     ;            BL: 0 -> PCM OUT data
1992                    <1>     ;              > 0 -> Invalid for now!
1993                    <1>     ;            ECX = 0 -> Get DMA Buffer Pointer
1994                    <1>     ;                EDX = Not Used
1995                    <1>     ;            ECX > 0 -> Byte count for buffer (EDX)
1996                    <1>     ;                EDX = Buffer Address (Virtual)
1997                    <1>     ;
1998                    <1>     ;       10/10/2017
1999                    <1>     ;       BH = 16 -> UPDATE DMA BUFFER DATA
2000                    <1>     ;                  (by using the Audio Buffer content)
2001                    <1>     ;            BL = 0 : Update dma half buffer in sequence
2002                    <1>     ;                     (automatic destination)
2003                    <1>     ;                 1 : Update 1st half of the dma buffer
2004                    <1>     ;                 2 : Update 2nd half of the dma buffer
2005                    <1>     ;                 3-FEh: Invalid!
2006                    <1>     ;                 FFh = Get current flag value
2007                    <1>     ;                      (Half buffer number -1)
2008                    <1>     ;
2009                    <1>     ;       24/05/2024
2010                    <1>     ;       BH = 17 -> GET VOLUME LEVEL
2011                    <1>     ;            BL:  0 = Master (Playback, Lineout) volume
2012                    <1>     ;                 1 = PCM out volume
2013                    <1>     ;
2014                    <1>     ; Outputs:
2015                    <1>     ;
2016                    <1>     ;       For BH = 0 -> Beep
2017                    <1>     ;           None
2018                    <1>     ;
2019                    <1>     ;       01/04/2017
2020                    <1>     ;
2021                    <1>     ;       For BH = 1 -> DETECT (& ENABLE) AUDIO DEVICE
2022                    <1>     ;           AH = 0 : PC SPEAKER
2023                    <1>     ;                1 : SOUND BLASTER 16
2024                    <1>     ;                2 : INTEL AC'97
2025                    <1>     ;                3 : VIA VT8237R (VT8233)
2026                    <1>     ;                4 : INTEL HDA
2027                    <1>     ;              5-FFh : unknown/invalid
2028                    <1>     ;           AL = mode status
2029                    <1>     ;                bit 0 = mono /stereo (1 = stereo)
2030                    <1>     ;                bit 1 = 8 bit / 16 bit (1 = 16 bit)
2031                    <1>     ;           04/06/2017
2032                    <1>     ;           EBX = PCI DEVICE/VENDOR ID (if >0)
2033                    <1>     ;                (BX = VENDOR ID)
```

```
2034                                 <1>     ;           (if CF = 1 -> Error code in EAX)
2035                                 <1>     ;
2036                                 <1>     ;       For BH = 2 -> ALLOCATE AUDIO BUFFER (for user)
2037                                 <1>     ;           EAX = Physical Address of the buffer
2038                                 <1>     ;           (if CF = 1 -> Error code in EAX)
2039                                 <1>     ;
2040                                 <1>     ;       For BH = 3 -> INITIALIZE AUDIO DEVICE
2041                                 <1>     ;           (if CF = 1 -> Error code in EAX)
2042                                 <1>     ;
2043                                 <1>     ;       For BH = 4 -> START TO PLAY
2044                                 <1>     ;           none (if CF = 1 -> Error code in EAX)
2045                                 <1>     ;
2046                                 <1>     ;       For BH = 5 -> PAUSE
2047                                 <1>     ;           none (if CF = 1 -> Error code in EAX)
2048                                 <1>     ;
2049                                 <1>     ;       For BH = 6 -> CONTINUE TO PLAY
2050                                 <1>     ;           none (if CF = 1 -> Error code in EAX)
2051                                 <1>     ;
2052                                 <1>     ;       For BH = 7 -> STOP
2053                                 <1>     ;           none (if CF = 1 -> Error code in EAX)
2054                                 <1>     ;
2055                                 <1>     ;       For BH = 8 -> RESET
2056                                 <1>     ;           none (if CF = 1 -> Error code in EAX)
2057                                 <1>     ;
2058                                 <1>     ;       For BH = 9 -> CANCEL (CALLBACK or S.R.B. SERVICE)
2059                                 <1>     ;           none (if CF = 1 -> Error code in EAX)
2060                                 <1>     ;
2061                                 <1>     ;       For BH = 10 -> DEALLOCATE AUDIO BUFFER (for user)
2062                                 <1>     ;           none (if CF = 1 -> Error code in EAX)
2063                                 <1>     ;
2064                                 <1>     ;       For BH = 11 -> SET VOLUME LEVEL
2065                                 <1>     ;           none (if CF = 1 -> Error code in EAX)
2066                                 <1>     ;
2067                                 <1>     ;       For BH = 12 -> DISABLE AUDIO DEVICE
2068                                 <1>     ;           none (if CF = 1 -> Error code in EAX)
2069                                 <1>     ;
2070                                 <1>     ;       12/05/2017
2071                                 <1>     ;       For BH = 13 -> MAP DMA BUFFER TO USER
2072                                 <1>     ;           EAX = Physical Address of the buffer
2073                                 <1>     ;           (if CF = 1 -> Error code in EAX)
2074                                 <1>     ;
2075                                 <1>     ;       04/06/2017
2076                                 <1>     ;       For BH = 14 -> GET AUDIO DEVICE INFO
2077                                 <1>     ;       (for BL = 0) ; 05/06/2017
2078                                 <1>     ;           EAX = IRQ Number in AL
2079                                 <1>     ;                 Audio Device Number in AH
2080                                 <1>     ;           EBX = DEV/VENDOR ID
2081                                 <1>     ;                (DDDDDDDDDDDDDDDDVVVVVVVVVVVVVVVV)
2082                                 <1>     ;           ECX = BUS/DEV/FN
2083                                 <1>     ;                (00000000BBBBBBBBDDDDDFFF00000000)
2084                                 <1>     ;           EDX = NABMBAR/NAMBAR (for AC97)
2085                                 <1>     ;                 (Low word, DX = NAMBAR address)
2086                                 <1>     ;           EDX = Base IO Addr (DX) for SB16 & VT8233
2087                                 <1>     ;           (if CF = 1 -> Error code in EAX)
2088                                 <1>     ;                         (ERR_DEV_NOT_RDY = 15)
2089                                 <1>     ;       (for BL = 1) -for AC97- ; 19/11/2023
2090                                 <1>     ;           EAX = Extended Audio ID (MX28) in AX
2091                                 <1>     ;            AX bit 0 - VRA bit
2092                                 <1>     ;            HW of EAX = PCM output sample rate (HZ)
2093                                 <1>     ;           EBX = VENDOR ID 1 (BX), VENDOR ID 2 (HW)
2094                                 <1>     ;           (if CF = 1 -> Error code in EAX)
2095                                 <1>     ;                         (ERR_DEV_NOT_RDY = 15)
2096                                 <1>     ;           Note: EAX & EBX = 0 (for SB16,VIA,HDA)
2097                                 <1>     ;
2098                                 <1>     ;       22/06/2017
2099                                 <1>     ;       For BH = 15 -> GET CURRENT SOUND DATA
2100                                 <1>     ;                      (for graphics)
2101                                 <1>     ;       (for BL = 0)
2102                                 <1>     ;        If ECX input is 0
2103                                 <1>     ;           EAX = DMA Buffer Current Position (Offset)
2104                                 <1>     ;        If ECX input >  0
2105                                 <1>     ;           EAX = Actual transfer count
2106                                 <1>     ;           (Sound samples will be copied from
2107                                 <1>     ;            Current DMA Buffer Position to EDX
2108                                 <1>     ;            virtual address as EAX bytes.)
2109                                 <1>     ;        ((If CF = 1 ->  Error code in EAX))
2110                                 <1>     ;
2111                                 <1>     ;       10/10/2017
2112                                 <1>     ;       For BH = 16 -> UPDATE DMA BUFFER DATA
2113                                 <1>     ;           EAX = 0, if the updated (or current)
2114                                 <1>     ;                    half buffer is DMA half buffer 1
2115                                 <1>     ;           EAX = 1, if the updated (or current)
2116                                 <1>     ;                    half buffer is DMA half buffer 2
2117                                 <1>     ;           (If CF = 1 -> Error code in EAX)
2118                                 <1>     ;
2119                                 <1>     ;       24/05/2024
2120                                 <1>     ;       For BH = 17 -> GET VOLUME LEVEL
2121                                 <1>     ;           IF BL input is 0
2122                                 <1>     ;              Master (Playback, Lineout) volume
2123                                 <1>     ;           If BL input is 1
2124                                 <1>     ;              PCM out volume
2125                                 <1>     ;           If BL input > 1
2126                                 <1>     ;              Invalid for now ; 24/05/2024
2127                                 <1>     ;
2128                                 <1>     ;           CL = Left Channel Volume (0 to 31 max)
2129                                 <1>     ;           CH = Right Channel Volume (0 to 31 max)
2130                                 <1>     ;
2131                                 <1>
2132 00012031 80FF12                <1>     cmp     bh, AUDIO1L/4
2133 00012034 0F8344AAFFFF          <1>     jnb     sysret
2134                                 <1>
2135 0001203A C0E702                <1>     shl     bh, 2 ; *4
2136 0001203D 0FB6F7                <1>     movzx   esi, bh
2137                                 <1>
2138                                 <1>     ; 22/04/2017
2139 00012040 31C0                  <1>     xor     eax, eax
2140 00012042 A3[348E0100]          <1>     mov     [u.r0], eax  ; 0
2141                                 <1>
2142 00012047 FF96[52200100]        <1>     call    dword [esi+AUDIO1]
2143                                 <1>     ;jc     error
2144 0001204D E92CAAFFFF            <1>     jmp     sysret
2145                                 <1>
2146 00012052 [BD230000]            <1> AUDIO1:     dd      _beep ; 12/02/2021
2147                                 <1>     ;dd     beep ; FUNCTION = 0 (bl = Duration Counter
2148                                 <1>                  ;                     cx = Frequency Divisor
2149 00012056 [9A200100]            <1>     dd      soundc_detect
2150 0001205A [2F210100]            <1>     dd      sound_alloc
2151 0001205E [F2210100]            <1>     dd      soundc_init
2152 00012062 [D6230100]            <1>     dd      sound_play
2153 00012066 [63240100]            <1>     dd      sound_pause
2154 0001206A [8D240100]            <1>     dd      sound_continue
2155 0001206E [B7240100]            <1>     dd      sound_stop
2156 00012072 [E2240100]            <1>     dd      soundc_reset
2157 00012076 [15250100]            <1>     dd      soundc_cancel
```

```
2158 0001207A [3B250100]          <1>     dd      sound_dalloc
2159 0001207E [66250100]          <1>     dd      sound_volume
2160 00012082 [A6250100]          <1>     dd      soundc_disable
2161 00012086 [24260100]          <1>     dd      sound_dma_map
2162 0001208A [92260100]          <1>     dd      soundc_info
2163 0001208E [16270100]          <1>     dd      sound_data
2164 00012092 [B8270100]          <1>     dd      sound_update
2165 00012096 [3A280100]          <1>     dd      sound_getvol ; 24/05/2024
2166                              <1> %if 0
2167                              <1>     dd      sound_dmaclear ; 29/12/2024
2168                              <1> %endif
2169                              <1>
2170                              <1> AUDIO1L     EQU     $ - AUDIO1
2171                              <1>
2172                              <1> soundc_detect:
2173                              <1>     ; FUNCTION = 1
2174                              <1>     ; bl = Audio device type number
2175                              <1>     ; (0 = pc speaker, 1 = sound blaster 16, 2 = intel ac97
2176                              <1>     ;  3 = via vt823x, 4 = intel HDA, 0FFh = any)
2177                              <1>
2178                              <1>     ; 04/06/2017
2179                              <1>     ;mov    ah, [audio_device]
2180                              <1>     ; 11/01/2025
2181 0001209A A0[B1880100]        <1>     mov     al, [audio_device]
2182 0001209F 80FBFF              <1>     cmp     bl, 0FFh ; get current audio device id
2183 000120A2 7408                <1>     je      short sysaudio0
2184                              <1>
2185                              <1>     ;and    ah, ah
2186                              <1>     ; 11/01/2025
2187 000120A4 20C0                <1>     and     al, al
2188 000120A6 741F                <1>     jz      short soundc_get_dev
2189                              <1>
2190                              <1>     ;cmp    ah, bl
2191                              <1>     ; 11/01/2025
2192 000120A8 38D8                <1>     cmp     al, bl
2193 000120AA 7568                <1>     jne     short soundc_dev_err
2194                              <1>
2195                              <1> sysaudio0:
2196                              <1>     ;mov    al, [audio_mode]
2197                              <1>     ; 11/01/2025
2198 000120AC 8A25[E9880100]      <1>     mov     ah, [audio_mode]
2199                              <1> sysaudio1:
2200 000120B2 A3[348E0100]        <1>     mov     [u.r0], eax
2201 000120B7 8B1D[BC880100]      <1>     mov     ebx, [audio_vendor] ; (DEVICE/VENDOR ID)
2202 000120BD 8B2D[308E0100]      <1>     mov     ebp, [u.usp]
2203 000120C3 895D10              <1>     mov     [ebp+16], ebx  ; ebx
2204 000120C6 C3                  <1>     retn
2205                              <1>
2206                              <1> soundc_get_dev:
2207                              <1>     ; 28/05/2017
2208                              <1>     ; 03/04/2017, 24/04/2017
2209 000120C7 C605[B0880100]00    <1>     mov     byte [audio_pci], 0
2210 000120CE 80FB03              <1>     cmp     bl, 3 ; VIA VT8233 (VT8237R) Audio Controller & AC97 Codec
2211                              <1>     ;jne    short soundc_get_dev_sb
2212                              <1>     ; 28/05/2017
2213 000120D1 7220                <1>     jb      short soundc_get_dev_sb
2214 000120D3 773F                <1>     ja      short soundc_dev_err  ; temporary (28/05/2017)
2215                              <1>     ;
2216 000120D5 E893180000          <1>     call    DetectVT8233
2217 000120DA 7238                <1>     jc      short soundc_dev_err
2218                              <1>     ; eax = 0
2219                              <1>
2220                              <1>     ;mov    ebx, [audio_vendor]
2221                              <1>     ; ebx = DEVICE/VENDOR ID
2222                              <1>     ;       DDDDDDDDDDDDDDDDVVVVVVVVVVVVVVVV
2223                              <1>
2224 000120DC B003                <1>     mov     al, 3  ; VIA VT8237R (VT3233) Audio Controller
2225 000120DE 88C4                <1>     mov     ah, al
2226                              <1>
2227                              <1> soundc_get_pci_dev_ok: ; 28/05/2017
2228 000120E0 FE05[B0880100]      <1>     inc     byte [audio_pci] ; = 1
2229                              <1> soundc_get_dev_ok:
2230                              <1>
2231                              <1> soundc_get_dev_sb16_ok:
2232 000120E6 A2[B1880100]        <1>     mov     [audio_device], al
2233 000120EB 8825[E9880100]      <1>     mov     [audio_mode], ah ; stereo (bit0), 16 bit (bit1) capability
2234 000120F1 EBBF                <1>     jmp     short sysaudio1
2235                              <1>
2236                              <1> soundc_get_dev_sb:
2237                              <1>     ; 24/04/2017
2238 000120F3 80FB01              <1>     cmp     bl, 1 ; Sound Blaster 16
2239 000120F6 750E                <1>     jne     short soundc_get_dev_ich ; 28/05/2017
2240                              <1>     ;
2241 000120F8 E8441D0000          <1>     call    DetectSB
2242 000120FD 7215                <1>     jc      short soundc_dev_err
2243 000120FF B801030000          <1>     mov     eax, 0301h ; Sound Blaster 16
2244 00012104 EBE0                <1>     jmp     short soundc_get_dev_sb16_ok
2245                              <1>
2246                              <1> soundc_get_dev_ich:
2247                              <1>     ; 28/05/2017
2248                              <1>     ;cmp    bl, 2 ; Intel AC'97 Audio Controller (ICH)
2249                              <1>     ;jne    short soundc_dev_err ; Temporary  (28/05/2017)
2250                              <1>     ;                            ; (Here will be modified just after
2251                              <1>     ;                            ; new sound card code will be ready!)
2252 00012106 E84D180000          <1>     call    DetectICH
2253 0001210B 7207                <1>     jc      short soundc_dev_err
2254                              <1>     ;
2255 0001210D B802030000          <1>     mov     eax, 0302h ; AC'97 (ICH)
2256 00012112 EBCC                <1>     jmp     short soundc_get_pci_dev_ok
2257                              <1>
2258                              <1> soundc_dev_err:
2259 00012114 B80F000000          <1>     mov     eax, ERR_DEV_NOT_RDY ; Device not ready !
2260                              <1>     ; 29/07/2022
2261                              <1>     ;sub    eax, eax
2262                              <1>     ;mov    al, ERR_DEV_NOT_RDY
2263 00012119 EB05                <1>     jmp     short sysaudio_err
2264                              <1>
2265                              <1> soundc_respond_err:
2266                              <1>     ; ERR_TIME_OUT ; 'time out !' error
2267 0001211B B819000000          <1>     mov     eax, ERR_DEV_NOT_RESP ; 'device not responding !' error
2268                              <1>     ; 29/07/2022
2269                              <1>     ;sub    eax, eax
2270                              <1>     ;mov    al, ERR_DEV_NOT_RESP
2271                              <1> sysaudio_err:
2272 00012120 A3[348E0100]        <1>     mov     [u.r0], eax
2273 00012125 A3[A08E0100]        <1>     mov     [u.error], eax
2274 0001212A E92FA9FFFF          <1>     jmp     error
2275                              <1>
2276                              <1> sound_alloc:
2277                              <1>     ; FUNCTION = 2
2278                              <1>     ; ecx = audio buffer size (in bytes)
2279                              <1>     ; edx = audio buffer address (virtual)
2280                              <1>     ; 28/01/2025
2281                              <1>     ; 27/01/2025 (BugFix)
```

```
                                <1>     ; 15/01/2025
                                <1>     ; 12/01/2025
                                <1>     ; 25/11/2023
                                <1>     ; 27/07/2020
                                <1>     ; 28/05/2027
                                <1>     ; 01/05/2017, 15/05/2017
                                <1>     ; 21/04/2017, 24/04/2017
0001212F 803D[B0880100]00       <1>     cmp     byte [audio_pci], 0
00012136 770F                   <1>     ja      short snd_alloc_0
                                <1>     ; Max. 64KB DMA buffer !!!
00012138 81F900800000           <1>     cmp     ecx, 32768
0001213E 760F                   <1>     jna     short snd_alloc_6
                                <1>     ; 25/11/2023
                                <1> sound_buff_error:
00012140 B82E000000             <1>     mov     eax, ERR_BUFFER ; Buffer error !
                                <1>     ; 29/07/2022
                                <1>     ;sub    eax, eax
                                <1>     ;mov    al, ERR_BUFFER
00012145 EBD9                   <1>     jmp     short sysaudio_err
                                <1> snd_alloc_0:
                                <1>     ; 25/11/2023
                                <1>     ; Max 128KB DMA buffer size (2 half buffers)
00012147 81F900000100           <1>     cmp     ecx, 65536
0001214D 77F1                   <1>     ja      short sound_buff_error
                                <1> snd_alloc_6:
                                <1>     ; 15/05/2017
                                <1>     ;cmp    ecx, 4096 ; PAGE_SIZE
                                <1>     ;jb     short sound_buff_error
                                <1>     ; 15/01/2025
0001214F 81F940010000           <1>     cmp     ecx, 320
00012155 72E9                   <1>     jb      short sound_buff_error
                                <1>     ;
                                <1>     ; 28/01/2025
                                <1>     ; 12/01/2025
                                <1>     ;mov    ebx, edx ; (new) virtual address of audio_buffer
                                <1>     ;
00012157 A1[C4880100]           <1>     mov     eax, [audio_buffer] ; audio buffer address (current)
0001215C 09C0                   <1>     or      eax, eax
                                <1>     ;jz     short snd_alloc_2
                                <1>     ; 12/01/2025
0001215E 743E                   <1>     jz      short snd_alloc_1 ; first time
                                <1>     ; audio buffer exists !
00012160 8A1D[858E0100]         <1>     mov     bl, [u.uno]
00012166 3A1D[DD880100]         <1>     cmp     bl, [audio_user]
                                <1>     ;jne    sndc_owner_error ; not owner !
                                <1>     ; 25/11/2023
                                <1>     ;je     short snd_alloc_7
                                <1>     ; 12/01/2025
0001216C 7405                   <1>     je      short snd_alloc_2
0001216E E9F3000000             <1>     jmp     sndc_owner_error ; not owner !
                                <1>
                                <1>     ; 12/01/2025 - TRDOS 386 v2.0.10
                                <1>     ; Important modification:
                                <1>     ;; Always deallocate user's audio_buffer address with
                                <1>     ;; buffer size and then allocate a new one
                                <1>     ;; (address range may be used before)
                                <1> snd_alloc_2:
                                <1>     ;mov    ecx, [audio_buff_size]
                                <1>     ; 28/01/2025
00012173 870D[CC880100]         <1>     xchg    ecx, [audio_buff_size] ; *
00012179 89C3                   <1>     mov     ebx, eax ; audio buffer address (current)
                                <1> ;snd_alloc_7:
                                <1>     ; 12/01/2025
                                <1>     ;cmp    eax, edx ; same virtual buffer address ?
                                <1>     ;jne    short snd_alloc_1
                                <1>     ;cmp    ecx, [audio_buff_size]
                                <1>     ;je     short snd_alloc_3 ; Nothing to do !
                                <1>     ;                         ; Buffer has been set already!
                                <1> ;snd_alloc_1:
                                <1>     ; 28/01/2025
                                <1>     ;push   ecx
0001217B 52                     <1>     push    edx
                                <1>     ; 12/01/2025
                                <1>     ;mov    ebx, eax ; audio buffer address (current)
                                <1>     ;mov    ecx, [audio_buff_size]
                                <1>     ; ebx = audio buffer address (virtual)
                                <1>     ; ecx = audio buffer size in bytes
0001217C E8103FFFFF             <1>     call    deallocate_user_pages
00012181 5A                     <1>     pop     edx
                                <1>     ;pop    ecx
                                <1>     ; 28/01/2025
00012182 8B0D[CC880100]         <1>     mov     ecx, [audio_buff_size] ; * ; requested size
00012188 31C0                   <1>     xor     eax, eax ; 0
0001218A A3[C4880100]           <1>     mov     [audio_buffer], eax  ; 0
0001218F A3[C8880100]           <1>     mov     [audio_p_buffer], eax  ; 0
00012194 A3[CC880100]           <1>     mov     [audio_buff_size], eax
00012199 A2[DD880100]           <1>     mov     [audio_user], al ; 0
                                <1>
                                <1>     ; 27/01/2025 (BugFix)
                                <1> snd_alloc_1:
                                <1>     ; 12/01/2025
                                <1> ;snd_alloc_2:
0001219E 89D3                   <1>     mov     ebx, edx
                                <1>     ; 01/05/2017
                                <1>     ;mov    edx, ~PAGE_OFF ; truncating page offsets
                                <1>     ;                      ; for aligning to page borders
                                <1>     ;;and   eax, edx
                                <1>     ;and    ebx, edx
                                <1>     ; 26/11/2023
                                <1>     ;and    ecx, edx
000121A0 81E300F0FFFF           <1>     and     ebx, ~PAGE_OFF
000121A6 89CA                   <1>     mov     edx, ecx
                                <1>     ; 15/05/2017
                                <1>     ; EAX = Beginning address (physical)
                                <1>     ; EAX = 0 -> Allocate mem block from the 1st proper aperture
                                <1>     ; ECX = Number of bytes to be allocated
000121A8 E88C3BFFFF             <1>     call    allocate_memory_block
000121AD 7291                   <1>     jc      short sound_buff_error
                                <1>
                                <1>     ; EAX = Physical address of the allocated memory block
                                <1>     ; ECX = Allocated bytes (as rounded up to page border)
                                <1>     ; EBX = Virtual address (as truncated to page border)
000121AF 52                     <1>     push    edx ; 26/11/2023
000121B0 50                     <1>     push    eax
000121B1 53                     <1>     push    ebx
000121B2 51                     <1>     push    ecx
000121B3 E8C13FFFFF             <1>     call    allocate_user_pages
000121B8 59                     <1>     pop     ecx
000121B9 5B                     <1>     pop     ebx
000121BA 58                     <1>     pop     eax
000121BB 5A                     <1>     pop     edx ; 26/11/2023
000121BC 722A                   <1>     jc      short snd_alloc_4 ; insufficient memory, buff error
                                <1>     ; eax = physical address of the user's audio buffer
                                <1>     ; ebx = virtual address of the user's audio buffer
```

```
                                <1>     ; 26/11/2023
                                <1>     ; ecx = allocated buffer size (in bytes)
                                <1>     ;               -rounded up to page boundary-
                                <1>     ; edx = requested buffer size (in bytes)
000121BE A3[C8880100]           <1>     mov     [audio_p_buffer], eax
000121C3 891D[C4880100]         <1>     mov     [audio_buffer], ebx
                                <1>     ; 26/11/2023
                                <1>     ;mov    ecx, edx
                                <1>     ;; 25/11/2023
                                <1>     ;cmp    ecx, 65536
                                <1>     ;jb     short snd_alloc_5
                                <1>     ;dec    ecx
                                <1>     ;; ecx = 65535
                                <1>     ;; (DMA half buffer's sample count must be < 65536)
                                <1> ;snd_alloc_5:
                                <1>     ;mov    [audio_buff_size], ecx
                                <1>     ; 26/11/2023
000121C9 8915[CC880100]         <1>     mov     [audio_buff_size], edx
                                <1>     ;
000121CF 8A15[858E0100]         <1>     mov     dl, [u.uno]
000121D5 8815[DD880100]         <1>     mov     [audio_user], dl
000121DB A3[348E0100]           <1>     mov     [u.r0], eax
                                <1> snd_alloc_3:
                                <1>     ; 27/07/2020
000121E0 C605[DC880100]00       <1>     mov     byte [audio_flag], 0 ; clear dma half buffer flag
                                <1>     ;
000121E7 C3                     <1>     retn
                                <1> snd_alloc_4:
                                <1>     ; 15/05/2017
                                <1>     ; EAX = Beginning address (physical)
                                <1>     ; ECX = Number of bytes to be deallocated
000121E8 E8663DFFFF             <1>     call    deallocate_memory_block
000121ED E94EFFFFFF             <1>     jmp     sound_buff_error ; insufficient memory, buff error
                                <1>
                                <1> soundc_init:
                                <1>     ; FUNCTION = 3
                                <1>     ; bl = method (0= s.r.b., 1= callback, 2= auto incr s.r.b.)
                                <1>     ; cl = signal response byte (initial or fixed) value
                                <1>     ; edx = signal response byte or callback address
                                <1>
                                <1>     ; 11/01/2025
                                <1>     ; 05/06/2024
                                <1>     ; 04/06/2024
                                <1>     ; 02/12/2023, 04/12/2022
                                <1>     ; 29/07/2022, 07/08/2022
                                <1>     ; 27/07/2020
                                <1>     ; 12/05/2017, 20/05/2017, 28/05/2027
                                <1>     ; 22/04/2017, 23/04/2017, 24/04/2017
                                <1>     ; 13/04/2017, 14/04/2017, 16/04/2017, 21/04/2017
                                <1>     ; 03/04/2017, 10/04/2017
                                <1>
000121F2 A0[B1880100]           <1>     mov     al, [audio_device]
000121F7 20C0                   <1>     and     al, al
000121F9 754A                   <1>     jnz     short sndc_init6
                                <1>     ;
000121FB C605[B0880100]00       <1>     mov     byte [audio_pci], 0
00012202 52                     <1>     push    edx
00012203 53                     <1>     push    ebx
00012204 51                     <1>     push    ecx
00012205 E8371C0000             <1>     call    DetectSB
0001220A 7213                   <1>     jc      short sndc_init8
0001220C 66B80103               <1>     mov     ax, 0301h ; Sound Blaster 16
00012210 EB1E                   <1>     jmp     short sndc_init7
                                <1>
                                <1> sndc_init11:
                                <1>     ; 28/05/2017
00012212 E841170000             <1>     call    DetectICH ; Detect AC'97 (ICH) Audio Controller
00012217 7217                   <1>     jc      short sndc_init7
00012219 66B80203               <1>     mov     ax, 0302h ; Intel AC'97 Audio Device
0001221D EB0B                   <1>     jmp     short sndc_init12 ; (PCI device)
                                <1>
                                <1> sndc_init8:
0001221F E849170000             <1>     call    DetectVT8233
                                <1>     ;jc     short sndc_init7
                                <1>     ; 29/07/2022
00012224 72EC                   <1>     jc      short sndc_init11 ; 28/05/2017
                                <1>
                                <1>     ; eax = 0
00012226 B003                   <1>     mov     al, 3   ; VIA VT8237R (VT8233) Audio Controller
00012228 88C4                   <1>     mov     ah, al
                                <1>
                                <1> sndc_init12:
0001222A FE05[B0880100]         <1>     inc     byte [audio_pci] ; = 1
                                <1> sndc_init7:
00012230 59                     <1>     pop     ecx
00012231 5B                     <1>     pop     ebx
00012232 5A                     <1>     pop     edx
                                <1>     ;jc     soundc_dev_err
                                <1>     ; 29/07/2022
00012233 7305                   <1>     jnc     short sndc_init14
00012235 E9DAFEFFFF             <1>     jmp     soundc_dev_err
                                <1> sndc_init14:
0001223A A2[B1880100]           <1>     mov     [audio_device], al
0001223F 8825[E9880100]         <1>     mov     [audio_mode], ah ; stereo (bit0), 16 bit (bit1) capability
                                <1>
                                <1> sndc_init6:
00012245 833D[C4880100]00       <1>     cmp     dword [audio_buffer], 0
                                <1>     ;jna    sound_buff_error
                                <1>     ; 29/07/2022
0001224C 7705                   <1>     ja      short sndc_init19
0001224E E9EDFEFFFF             <1>     jmp     sound_buff_error ; 07/08/2022
                                <1> sndc_init19:
00012253 A0[858E0100]           <1>     mov     al, [u.uno]
00012258 8A25[DD880100]         <1>     mov     ah, [audio_user]
0001225E 08E4                   <1>     or      ah, ah
00012260 7418                   <1>     jz      short sndc_init0
00012262 38E0                   <1>     cmp     al, ah
00012264 7419                   <1>     je      short sndc_init1
                                <1>
                                <1> sndc_owner_error:
00012266 B80B000000             <1>     mov     eax, ERR_NOT_OWNER ; 'permission denied !' error
                                <1>     ; 29/07/2022
                                <1>     ;sub    eax, eax
                                <1>     ;mov    al, ERR_NOT_OWNER
                                <1> sndc_perm_error:
0001226B A3[348E0100]           <1>     mov     [u.r0], eax
00012270 A3[A08E0100]           <1>     mov     [u.error], eax
00012275 E9E4A7FFFF             <1>     jmp     error
                                <1> sndc_init0:
0001227A A2[DD880100]           <1>     mov     [audio_user], al
                                <1> sndc_init1:
0001227F 8915[E0880100]         <1>     mov     [audio_cb_addr], edx
00012285 881D[DE880100]         <1>     mov     [audio_cb_mode], bl
0001228B 880D[DF880100]         <1>     mov     [audio_srb], cl
```

```
                                <1>
                                <1>     ; 27/07/2020
                                <1>     ;mov    byte [audio_flag], 0  ; clear dma half buffer flag
                                <1>
                                <1>     ; 24/04/2017
00012291 803D[B1880100]03       <1>     cmp     byte [audio_device], 3 ; VT8233 (VT8237R)
00012298 7435                   <1>     je      short sndc_init9
                                <1>     ;ja     short soundc_respond_err ; temporary (28/05/2017)
0001229A 803D[B1880100]01       <1>     cmp     byte [audio_device], 1 ; SB 16
000122A1 7510                   <1>     jne     short sndc_init13
000122A3 BB[53400100]           <1>     mov     ebx, sb16_int_handler
                                <1>     ; Note: 'SbInit' is at 'Start to Play' stage
                                <1>     ; 20/05/2017
000122A8 66C705[EC880100]08-    <1>     mov     word [audio_master_volume], 0808h ; 2/8
000122B0 08                     <1>
000122B1 EB34                   <1>     jmp     short sndc_init10
                                <1> sndc_init13:
                                <1>     ; 28/05/2017
000122B3 803D[B1880100]02       <1>     cmp     byte [audio_device], 2 ; AC 97 (ICH)
                                <1>     ;jne    soundc_respond_err ; temporary (28/05/2017)
                                <1>     ; 29/07/2022
000122BA 7405                   <1>     je      short sndc_init16
                                <1> sndc_init15:
000122BC E95AFEFFFF             <1>     jmp     soundc_respond_err
                                <1> sndc_init16:
000122C1 E8D51E0000             <1>     call    ac97_codec_config
                                <1>     ;jc     soundc_respond_err ; codec error !
                                <1>     ; 29/07/2022
000122C6 72F4                   <1>     jc      short sndc_init15
                                <1>
000122C8 BB[35440100]           <1>     mov     ebx, ac97_int_handler
000122CD EB18                   <1>     jmp     short sndc_init10
                                <1>
                                <1> sndc_init9:
                                <1>     ;call   reset_codec
                                <1>     ;; eax = 1
                                <1>     ;call   codec_io_w16 ; w32
000122CF E8ED170000             <1>     call    init_codec ; 28/05/2017
                                <1>     ;jc     soundc_respond_err ; codec error !
                                <1>     ; 29/07/2022
000122D4 72E6                   <1>     jc      short sndc_init15
                                <1>
000122D6 E8FB190000             <1>     call    channel_reset
                                <1>
                                <1>     ; setup the Codec (actually mixer registers)
000122DB E824190000             <1>         call    codec_config  ; unmute codec, set rates.
                                <1>     ;jc     soundc_respond_err ; codec error !
                                <1>     ; 29/07/2022
000122E0 72DA                   <1>     jc      short sndc_init15
                                <1>
000122E2 BB[6A3C0100]           <1>     mov     ebx, vt8233_int_handler
                                <1> sndc_init10:
                                <1>     ; 13/04/2017
000122E7 A0[B2880100]           <1>     mov     al, [audio_intr] ; IRQ number
000122EC E833FDFFFF             <1>     call    set_dev_IRQ_service
                                <1>
                                <1>     ; SETUP (audio) INTERRUPT CALLBACK SERVICE
000122F1 8A1D[B2880100]         <1>     mov     bl, [audio_intr] ; IRQ number
000122F7 8A3D[DE880100]         <1>     mov     bh, [audio_cb_mode]
000122FD FEC7                   <1>     inc     bh  ; 1 = Signal Response Byte method (fixed value)
                                <1>                 ; 2 = Callback service method
                                <1>                 ; 3 = Auto Increment S.R.B. method
000122FF 8A0D[DF880100]         <1>     mov     cl, [audio_srb]
00012305 8B15[E0880100]         <1>     mov     edx, [audio_cb_addr]
0001230B A0[DD880100]           <1>     mov     al, [audio_user]
                                <1>     ; 14/04/2017
00012310 E83C050000             <1>     call    set_irq_callback_service
                                <1>     ; 16/04/2017
00012315 A3[348E0100]           <1>     mov     [u.r0], eax
                                <1>     ;jnc    sysret
0001231A 7316                   <1>     jnc     short sndc_init2 ; 21/04/2017
                                <1>     ;
0001231C A3[A08E0100]           <1>     mov     dword [u.error], eax
                                <1>
00012321 A0[B2880100]           <1>     mov     al, [audio_intr] ; IRQ number
00012326 31DB                   <1>     xor     ebx, ebx ; reset IRQ handler address
00012328 E8F7FCFFFF             <1>     call    set_dev_IRQ_service
                                <1>
0001232D E92CA7FFFF             <1>     jmp     error
                                <1>
                                <1> sndc_init2:
                                <1>     ; 21/04/2017
00012332 8B0D[CC880100]         <1>     mov     ecx, [audio_buff_size] ; audio buffer size
                                <1>     ; 05/06/2024
                                <1>     ;shl    ecx, 1 ; *2
                                <1>
                                <1>     ; 05/06/2024
00012338 A0[B1880100]           <1>     mov     al, [audio_device]
0001233D 3C01                   <1>     cmp     al, 1 ; SB16
0001233F 760A                   <1>     jna     short sndc_init22
00012341 80E1FE                 <1>     and     cl, ~1 ; truncated for word alignment
00012344 3C03                   <1>     cmp     al, 3 ; VT8233
00012346 7303                   <1>     jnb     short sndc_init22
                                <1>     ; al = 2 ; AC'97
00012348 80E1F8                 <1>     and     cl, ~7 ; truncated for 8 byte (8x) alignment
                                <1> sndc_init22:
                                <1>     ; 05/06/2024
0001234B 890D[D8880100]         <1>     mov     [dma_hbuff_size], ecx  ; DMA half buffer size
00012351 D1E1                   <1>     shl     ecx, 1  ; add ecx, ecx ; * 2
                                <1>
                                <1>     ;;;
                                <1>     ; 04/06/2024
00012353 81C1FF0F0000           <1>     add     ecx, 4095  ; PAGE_SIZE - 1
00012359 6681E100F0             <1>     and     cx, ~4095  ; ~PAGE_OFF
                                <1>     ; ecx = page border aligned DMA buffer size (required) (*)
                                <1>     ;;;
                                <1>
0001235E A1[D0880100]           <1>     mov     eax, [audio_dma_buff]
00012363 21C0                   <1>     and     eax, eax
00012365 741C                   <1>     jz      short sndc_init3 ; no need to compare dma buff size
                                <1>
00012367 8B15[D4880100]         <1>     mov     edx, [audio_dmabuff_size] ; dma buffer size
0001236D 39D1                   <1>     cmp     ecx, edx
0001236F 742B                   <1>     je      short sndc_init5
                                <1>
                                <1>     ; 04/12/2023
00012371 3D00000500             <1>     cmp     eax, sb16_dma_buffer  ; reserved buffer ?
00012376 7409                   <1>     je      short sndc_init20 ; it isn't an allocated mem buff
                                <1>
00012378 87D1                   <1>     xchg    ecx, edx
                                <1>     ; 26/11/2023
                                <1>     ; round up (always -rounded up- page count is allocated)
                                <1>     ; ((so deallocation must be done for the rounded up value))
                                <1>     ;add    ecx, PAGE_SIZE - 1   ; 4095
```

```
                                <1>     ;call   deallocate_memory_block
                                <1>     ; 04/06/2024
                                <1>     ;call   deallocate_memory_block_x
                                <1>     ;               ; deallocate ((ecx+4095)>>12) pages
0001237A E8D43BFFFF             <1>     call    deallocate_memory_block
                                <1>
0001237F 87CA                   <1>     xchg    edx, ecx
                                <1> sndc_init20:
00012381 31C0                   <1>     xor     eax, eax
                                <1> sndc_init3:
                                <1>     ; 05/06/2024
00012383 890D[D4880100]         <1>     mov     [audio_dmabuff_size], ecx ; (*)
                                <1>     ; 12/05/2017
00012389 803D[B1880100]01       <1>     cmp     byte [audio_device], 1 ; SB 16
00012390 750B                   <1>     jne     short sndc_init4
00012392 C705[D0880100]0000-    <1>     mov     dword [audio_dma_buff], sb16_dma_buffer
0001239A 0500                   <1>
                                <1>     ; 05/06/2024
                                <1>     ;mov    dword [audio_dmabuff_size], 65536
                                <1>     ;xor    eax, eax
                                <1>     ;mov    [u.r0], eax ; 0 = no error, successful
                                <1> sndc_init5: ; 29/07/2022
0001239C C3                     <1>     retn
                                <1>
                                <1> sndc_init4:
                                <1>     ; 02/12/2023 - TRDOS 386 v2.0.7
0001239D 81F900000100           <1>     cmp     ecx, 65536
000123A3 7707                   <1>     ja      short sndc_init21
000123A5 B800000500             <1>     mov     eax, sb16_dma_buffer ; use already reserved buffer
000123AA EB0C                   <1>     jmp     short sndc_init17
                                <1>
                                <1> sndc_init21:
                                <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)
000123AC E88839FFFF             <1>     call    allocate_memory_block
                                <1>     ;jc     sound_buff_error
                                <1>     ; 29/07/2022
000123B1 7305                   <1>     jnc     short sndc_init17
000123B3 E988FDFFFF             <1>     jmp     sound_buff_error
                                <1>
                                <1> sndc_init17:        ; 29/07/2022
                                <1>     ; set dma buffer address and size parameters
000123B8 A3[D0880100]           <1>     mov     [audio_dma_buff], eax ; dma buffer address
                                <1>     ; 05/06/2024
                                <1>     ;mov    [audio_dmabuff_size], ecx ; dma buffer size
                                <1>
                                <1>     ;;;
                                <1>     ; 05/06/2024
000123BD 8B0D[D8880100]         <1>     mov     ecx, [dma_hbuff_size] ; DMA half (1-2) buffer size
                                <1>     ; 04/06/2024
                                <1>     ;mov    ecx, [audio_buff_size] ; audio buffer size in bytes
                                <1>     ;;;
                                <1>
                                <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:
000123C3 803D[B1880100]03       <1>     cmp     byte [audio_device], 3
                                <1>     ;jne    short sndc_init5
                                <1>     ; 29/07/2022
000123CA 7505                   <1>     jne     short sndc_init18 ; 28/05/2017
                                <1>
                                <1> ; 29/07/2022
                                <1> ;   call    set_vt8233_bdl
                                <1> ;sndc_init5:
                                <1> ;   ;sub    eax, eax ; 0
                                <1> ;   ;mov    [u.r0], eax ; 0 = no error, successful
                                <1> ;   retn
                                <1>
                                <1>     ; 05/06/2024
                                <1>     ;;;
                                <1>     ; 04/06/2024 (use truncated buffer size for BDL setup)
                                <1>     ; NOTE: Round up adds spurious bytes (noise) to the DMA buff;
                                <1>     ;       so, round down the size is better than the round up.
                                <1>     ;       ((BDL/SGD feature needs word aligned buffer address))
                                <1>     ;
                                <1>     ;mov    ecx, [audio_buff_size] ; audio buffer size in bytes
                                <1>     ;and    cl, ~1
                                <1>     ; 05/06/2024
                                <1>     ;mov    [dma_hbuff_size], ecx
                                <1>     ; ecx = DMA half buffer size as word aligned
                                <1>     ;       (in fact, half buffer is one of the two DMA buffers)
                                <1>     ;;;
                                <1>
000123CC E93F190000             <1>     jmp     set_vt8233_bdl
                                <1>
                                <1> sndc_init18:
                                <1>     ;call   set_ac97_bdl
                                <1>     ;;jmp   short sndc_init5
                                <1>     ;retn
                                <1>
                                <1>     ; 05/06/2024
                                <1>     ;;;
                                <1>     ;;;
                                <1>     ; 04/06/2024 (use truncated buffer size for BDL setup)
                                <1>     ; NOTE: Round up adds spurious bytes (noise) to the DMA buff;
                                <1>     ;       so, round down the size is better than the round up.
                                <1>     ;       ((BDL feature needs 8 byte aligned buffer address))
                                <1>     ;
                                <1>     ;mov    ecx, [audio_buff_size] ; audio buffer size in bytes
                                <1>     ;and    cl, ~7
                                <1>     ; 05/06/2024
                                <1>     ;mov    [dma_hbuff_size], ecx
                                <1>     ; ecx = DMA half buffer size as truncated for 8 byte alignment
```

```
2776                                <1>     ;       (in fact, half buffer is one of the two DMA buffers)
2777                                <1>     ;;;
2778                                <1>
2779                                <1>     ; 29/07/2022
2780 000123D1 E96B1F0000           <1>     jmp     set_ac97_bdl
2781                                <1>
2782                                <1> sound_play:
2783                                <1>     ; FUNCTION = 4
2784                                <1>     ; bl = Mode
2785                                <1>     ;       bit 0 = mono/stereo (1 = stereo)
2786                                <1>     ;       bit 1 = 8 bit / 16 bit (1 = 16 bit)
2787                                <1>     ; cx = Sampling Rate (Hz)
2788                                <1>
2789                                <1>     ; 13/06/2017
2790                                <1>     ; Note: Even if Mode bits are not 11b,
2791                                <1>     ;       AC'97 Audio Controller (&Codec)
2792                                <1>     ;       will play audio samples as 16 bit, stereo
2793                                <1>     ;       samples.
2794                                <1>     ;       (Program must fill the audio buffer
2795                                <1>     ;       as required; 8 bit samples must be converted
2796                                <1>     ;       to 16 bit samples and mono samples must be
2797                                <1>     ;       converted to stereo samples...)
2798                                <1>
2799                                <1>     ; 19/12/2024 (BugFix)
2800                                <1>     ; 05/06/2024
2801                                <1>     ; 04/06/2024
2802                                <1>     ; 30/07/2022
2803                                <1>     ; 28/07/2020
2804                                <1>     ; 27/07/2020
2805                                <1>     ; 28/05/2017
2806                                <1>     ; 15/05/2017, 20/05/2017
2807                                <1>     ; 21/04/2017, 24/04/2017
2808                                <1>     ; ... device check at first
2809 000123D6 A0[B1880100]         <1>     mov     al, [audio_device]
2810 000123DB 08C0                 <1>     or      al, al ; 0 ; pc speaker or invalid
2811                                <1>     ;jz     beeper_gfx ; 'video.s' ; temporary
2812                                <1>     ; 30/07/022
2813 000123DD 7505                 <1>     jnz     short snd_play_7
2814 000123DF E9EAFFFEFF           <1>     jmp     beeper_gfx
2815                                <1>
2816                                <1> snd_play_7:
2817                                <1> ;   cmp     al, 3 ; VIA VT 8237R (vt8233)
2818                                <1> ;   je      short snd_play_1
2819                                <1> ;   cmp     al, 1 ; SB 16
2820                                <1> ;   jne     soundc_dev_err ; temporary !
2821                                <1> ;snd_play_0:
2822                                <1>     ; ... buffer & (buffer) owner check at second
2823 000123E4 833D[C4880100]00     <1>     cmp     dword [audio_buffer], 0
2824                                <1>     ;jna    sound_buff_error
2825                                <1>     ; 30/07/2022
2826 000123EB 763C                 <1>     jna     short snd_play_4 ; jmp sound_buff_error
2827 000123ED A0[858E0100]         <1>     mov     al, [u.uno]
2828 000123F2 3A05[DD880100]       <1>     cmp     al, [audio_user]
2829                                <1>     ;jne    sndc_owner_error
2830                                <1>     ; 30/07/2022
2831 000123F8 7405                 <1>     je      short snd_play_3
2832 000123FA E967FEFFFF           <1>     jmp     sndc_owner_error
2833                                <1> snd_play_3:
2834 000123FF 66890D[E6880100]     <1>     mov     [audio_freq], cx ; sample frequency (Hertz)
2835 00012406 88D8                 <1>     mov     al, bl
2836 00012408 2401                 <1>     and     al, 1 ; mono/stereo (1= stereo)
2837 0001240A FEC0                 <1>     inc     al ; channels
2838 0001240C A2[E5880100]         <1>     mov     [audio_stmo], al ; sound channels (1 or 2)
2839 00012411 B008                 <1>     mov     al, 8
2840 00012413 F6C302               <1>     test    bl, 2 ; bits per sample (1= 16 bit)
2841 00012416 7402                 <1>     jz      short snd_play_bps
2842 00012418 D0E0                 <1>     shl     al, 1
2843                                <1> snd_play_bps:
2844 0001241A A2[E4880100]         <1>     mov     [audio_bps], al
2845                                <1>
2846                                <1>     ; Transfer ring 3 (user's) audio buffer content to dma buffer
2847 0001241F 8B3D[D0880100]       <1>     mov     edi, [audio_dma_buff] ; dma buffer (ring 0)
2848 00012425 09FF                 <1>     or      edi, edi
2849                                <1>     ;jz     sound_buff_error
2850                                <1>     ; 30/07/2022
2851 00012427 7505                 <1>     jnz     short snd_play_5
2852                                <1> snd_play_4:
2853 00012429 E912FDFFFF           <1>     jmp     sound_buff_error
2854                                <1> snd_play_5:
2855                                <1>     ; 27/07/2020
2856 0001242E 8B35[C8880100]       <1>     mov     esi, [audio_p_buffer] ; physical address (ring 3)
2857                                <1>     ;mov    ecx, [audio_buff_size] ; 15/05/2017
2858                                <1>
2859                                <1> ; 04/06/2024
2860                                <1> %if 0
2861                                <1>     mov     ecx, [audio_dmabuff_size] ; 27/07/2020
2862                                <1>     ;or     ecx, ecx
2863                                <1>     ;jz     sound_buff_error
2864                                <1>     ; 28/07/2020
2865                                <1>     shr     ecx, 1  ; dma half buffer size
2866                                <1> %else
2867                                <1>     ; 05/06/2024
2868 00012434 8B0D[D8880100]       <1>     mov     ecx, [dma_hbuff_size]  ; DMA half buffer size
2869                                <1>
2870                                <1>     ; 04/06/2024
2871                                <1>     ;mov    ecx, [audio_buff_size]
2872                                <1>     ; 19/12/2024 (BugFix)
2873 0001243A A0[B1880100]         <1>     mov     al, [audio_device]
2874                                <1>     ;
2875                                <1>     ;cmp    al, 1 ; Sound Blaster 16
2876                                <1>     ;je     short snd_play_8
2877                                <1>     ;and    cl, ~1
2878                                <1>     ;cmp    al, 3 ; VT8233 (VT8237R)
2879                                <1>     ;je     short snd_play_8
2880                                <1>     ; AC'97
2881                                <1>     ;and    cl, ~7
2882                                <1> snd_play_8:
2883                                <1> %endif
2884                                <1>
2885 0001243F 8035[DC880100]01     <1>     xor     byte [audio_flag], 1 ;  0 -> 1, 1 -> 0
2886 00012446 7502                 <1>     jnz     short snd_play_0 ; [audio_flag] = 1
2887                                <1>                              ; fill dma half buffer 1
2888                                <1>     ; [audio_flag] = 0
2889                                <1>
2890                                <1>     ; fill dma half buffer 2
2891 00012448 01CF                 <1>     add     edi, ecx
2892                                <1>
2893                                <1> snd_play_0:
2894                                <1>     ; 05/06/2024
2895                                <1>     ;;rep   movsb
2896                                <1>     ;shr    ecx, 2 ; convert byte count to dword count
2897                                <1>     ;rep    movsd  ;
2898 0001244A F3A4                 <1>     rep     movsb  ; SB16, AC97, VT8233
2899                                <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
                                <1>     ;mov    al, [audio_device]
                                <1>     ; 04/06/2024
                                <1>     ; al = [audio_device]
                                <1>
0001244C 3C03                  <1>     cmp     al, 3 ; VT8233 (VT8237R)
0001244E 7409                  <1>     je      short snd_play_1
00012450 3C01                  <1>     cmp     al, 1 ; Sound Blaster 16
00012452 750A                  <1>     jne     short snd_play_2  ; 28/05/2017
                                <1>
                                <1> ; 30/07/2022
                                <1> ;   call    SbInit_play
                                <1> ;   ;jc     soundc_respond_err
                                <1> ;   ;retn
                                <1> ;   ; 30/07/2022
                                <1> ;   jnc     short snd_play_6  ; retn
                                <1> ;   jmp     soundc_respond_err
                                <1>
                                <1>     ; 30/07/2022
00012454 E9A81A0000             <1>     jmp     SbInit_play    ; sb16_start_play
                                <1>
                                <1> snd_play_1:
                                <1>     ;call   vt8233_start_play
                                <1>     ;retn
                                <1>     ; 30/07/2022
00012459 E9E6180000             <1>     jmp     vt8233_start_play
                                <1>
                                <1> snd_play_2:
                                <1>     ; 28/05/2017
                                <1>     ;cmp    al, 2 ; AC'97
0001245E E90F1F0000             <1>     ;jne    short snd_play_3
                                <1>
                                <1>     ;call   ac97_start_play
                                <1>     ;retn
                                <1>     ; 30/07/2022
                                <1>     jmp     ac97_start_play
                                <1>
                                <1> ;snd_play_3:
                                <1> ;   ;call   hda_start_play
                                <1> ;   retn
                                <1>     ; 30/07/2022
                                <1>     ;jmp    hda_start_play
                                <1>
                                <1> ; 04/06/2024
                                <1> ;snd_play_6:
                                <1> ;   ; 30/07/2022
                                <1> ;   retn
                                <1>
                                <1> sound_pause:
                                <1>     ; FUNCTION = 5
                                <1>     ; Pause
                                <1>     ; 28/05/2017
                                <1>     ; 24/04/2017
                                <1>     ; 22/04/2017
00012463 E82B030000             <1>     call    snd_dev_check
00012468 7277                  <1>     jc      short snd_nothing ; temporary.
0001246A E831030000             <1>     call    snd_buf_check
0001246F 7270                  <1>     jc      short snd_nothing ; temporary.
00012471 A0[B1880100]           <1>     mov     al, [audio_device]
00012476 3C03                  <1>     cmp     al, 3 ; VIA VT 8237R (vt8233)
00012478 7409                  <1>     je      short snd_pause_1
0001247A 3C01                  <1>     cmp     al, 1 ; Sound Blaster 16
0001247C 750A                  <1>     jne     short snd_pause_2 ; 28/05/2017
0001247E E9511C0000             <1>     jmp     sb16_pause
                                <1> snd_pause_1:
00012483 E971190000             <1>     jmp     vt8233_pause
                                <1> snd_pause_2:
                                <1>     ; 28/05/2017
                                <1>     ;cmp    al, 2 ; AC'97
                                <1>     ;jne    short snd_nothing ; temporary.
00012488 E9E11F0000             <1>     jmp     ac97_pause
                                <1>
                                <1> sound_continue:
                                <1>     ; FUNCTION = 6
                                <1>     ; Continue to play
                                <1>     ; 28/05/2017
                                <1>     ; 22/04/2017
0001248D E801030000             <1>     call    snd_dev_check
00012492 724D                  <1>     jc      short snd_nothing ; temporary.
00012494 E807030000             <1>     call    snd_buf_check
00012499 7246                  <1>     jc      short snd_nothing ; temporary.
0001249B A0[B1880100]           <1>     mov     al, [audio_device]
000124A0 3C03                  <1>     cmp     al, 3 ; VIA VT 8237R (vt8233)
000124A2 7409                  <1>     je      short snd_cont_1
000124A4 3C01                  <1>     cmp     al, 1 ; Sound Blaster 16
000124A6 750A                  <1>     jne     short snd_cont_2 ; 28/05/2017
000124A8 E9491C0000             <1>     jmp     sb16_continue
                                <1> snd_cont_1:
000124AD E9F9180000             <1>     jmp     vt8233_play
                                <1> snd_cont_2:
                                <1>     ; 28/05/2017
                                <1>     ;cmp    al, 2 ; AC'97
                                <1>     ;;jne   short snd_nothing ; temporary.
                                <1>     ; 30/07/2022
                                <1>     ;jne    short snd_cont_3
000124B2 E9F81E0000             <1>     jmp     ac97_play
                                <1> ;snd_cont_3:
                                <1>     ;jmp    hda_play
                                <1>
                                <1> sound_stop:
                                <1>     ; FUNCTION = 7
                                <1>     ; Stop playing
                                <1>     ; 30/07/2022
                                <1>     ; 28/05/2017
                                <1>     ; 24/05/2017
                                <1>     ; 21/04/2017, 22/04/2017, 24/04/2017
000124B7 E8D7020000             <1>     call    snd_dev_check
000124BC 7223                  <1>     jc      short snd_nothing ; temporary.
                                <1>     ;call   snd_buf_check
000124BE E8E6020000             <1>     call    snd_user_check ; 24/05/2017
000124C3 721C                  <1>     jc      short snd_nothing ; temporary.
                                <1>
000124C5 A0[B1880100]           <1>     mov     al, [audio_device]
000124CA 3C03                  <1>     cmp     al, 3 ; VIA VT 8237R (vt8233)
                                <1>     ;je     vt8233_stop
                                <1>     ;; 28/05/2017
                                <1>     ;;ja    short snd_nothing
```

```
                                <1>     ; 30/07/2022
000124CC 7505                   <1>     jne     short snd_stop_1
000124CE E934180000             <1>     jmp     vt8233_stop
                                <1> snd_stop_1:
000124D3 3C01                   <1>     cmp     al, 1 ; Sound Blaster 16
                                <1>     ;je     sb16_stop
                                <1>     ; 30/07/2022
000124D5 7505                   <1>     jne     short snd_stop_2
000124D7 E93C1C0000             <1>     jmp     sb16_stop
                                <1> snd_stop_2:
                                <1>     ;cmp    al, 2
                                <1>     ;;je    short ac97_stop
                                <1>     ; 30/07/2022
                                <1>     ;jne    short snd_stop_3
000124DC E9841F0000             <1>     jmp     ac97_stop ; temporary.
                                <1> ;snd_stop_3:
                                <1>     ;jmp    hda_stop
                                <1>
                                <1> sndc_cancel_ok:
                                <1>     ; 30/07/2022
                                <1> sndc_reset_ok:
                                <1>     ; 30/07/2022
                                <1> snd_nothing:
                                <1>     ; 21/04/2017
000124E1 C3                     <1>     retn
                                <1>
                                <1> soundc_reset:
                                <1>     ; FUNCTION = 8
                                <1>     ; Reset Audio Controller
                                <1>     ; 30/07/2022
                                <1>     ; 28/05/2017
                                <1>     ; 22/04/2017
000124E2 E8AC020000             <1>     call    snd_dev_check
000124E7 72F8                   <1>     jc      short snd_nothing ; temporary.
000124E9 E8B2020000             <1>     call    snd_buf_check
000124EE 72F1                   <1>     jc      short snd_nothing ; temporary.
                                <1>
000124F0 A0[B1880100]           <1>     mov     al, [audio_device]
                                <1>     ; 30/07/2022
000124F5 3C03                   <1>     cmp     al, 3 ; VIA VT 8237R (vt8233)
000124F7 7207                   <1>     jb      short sndc_reset_1
                                <1>     ;je     vt8233_reset
000124F9 77E6                   <1>     ja      short snd_nothing ; temporary.
                                <1>     ;;ja    hda_reset
                                <1>     ;ja     short sndc_reset_3
                                <1>     ; 30/07/2022
000124FB E905190000             <1>     jmp     vt8233_reset
                                <1> sndc_reset_1:
00012500 3C01                   <1>     cmp     al, 1 ; Sound Blaster 16
                                <1>     ;jne    ac97_reset
                                <1>     ; 30/07/2022
00012502 750C                   <1>     jne     short sndc_reset_2
00012504 E8601C0000             <1>     call    sb16_reset
                                <1>     ;jc     soundc_respond_err
                                <1>     ;retn
                                <1>     ; 30/07/2022
00012509 73D6                   <1>     jnc     short sndc_reset_ok
0001250B E90BFCFFFF             <1>     jmp     soundc_respond_err
                                <1> sndc_reset_2:
                                <1>     ; 30/07/2022
                                <1>     ;cmp    al, 2
                                <1>     ;ja     short sndc_reset_3
00012510 E915200000             <1>     jmp     ac97_reset
                                <1> ;sndc_reset_3:
                                <1>     ;jmp    hda_reset
                                <1>
                                <1> soundc_cancel:
                                <1>     ; FUNCTION = 9
                                <1>     ; Cancel audio callback service
                                <1>     ; 30/07/2022
                                <1>     ; 22/04/2017
00012515 A0[DD880100]           <1>     mov     al, [audio_user]
0001251A 3A05[858E0100]         <1>     cmp     al, [u.uno]
00012520 75BF                   <1>     jne     short snd_nothing
                                <1>     ; RESET (audio) INTERRUPT CALLBACK SERVICE
00012522 8A1D[B2880100]         <1>     mov     bl, [audio_intr] ; IRQ number
00012528 A0[858E0100]           <1>     mov     al, [u.uno]
0001252D 28FF                   <1>     sub     bh, bh ; 0 ; unlink IRQ from user service
0001252F E81D030000             <1>     call    set_irq_callback_service
                                <1>     ;jc     sndc_perm_error ; 'permission denied' error
                                <1>     ;retn
                                <1>     ; 30/07/2022
00012534 73AB                   <1>     jnc     short sndc_cancel_ok
00012536 E930FDFFFF             <1>     jmp     sndc_perm_error
                                <1>
                                <1> sound_dalloc:
                                <1>     ; FUNCTION = 10
                                <1>     ; Deallocate (ring 3) audio buffer
                                <1>     ; 22/04/2017
0001253B A0[DD880100]           <1>     mov     al, [audio_user]
00012540 3A05[858E0100]         <1>     cmp     al, [u.uno]
00012546 7599                   <1>     jne     short snd_nothing
00012548 8B1D[C4880100]         <1>     mov     ebx, [audio_buffer]
                                <1>     ;or     ebx, ebx
                                <1>     ;jz     short snd_nothing
0001254E 8B0D[CC880100]         <1>     mov     ecx, [audio_buff_size]
00012554 E8383BFFFF             <1>     call    deallocate_user_pages
00012559 31C0                   <1>     xor     eax, eax
0001255B A3[C4880100]           <1>     mov     [audio_buffer], eax ; 0
00012560 A2[DD880100]           <1>     mov     [audio_user], al ; 0
                                <1> ;sndc_cancel_ok:
00012565 C3                     <1>     retn
                                <1>
                                <1> sound_volume:
                                <1>     ; FUNCTION = 11
                                <1>     ; Set sound volume level
                                <1>     ; 23/05/2024
                                <1>     ; 30/07/2022
                                <1>     ; 28/05/2017
                                <1>     ; 20/05/2017
                                <1>     ; 22/04/2017, 24/04/2017
                                <1>     ; bl = component
                                <1>     ;              0 = master/playback/lineout volume
                                <1>     ;              1 = PCM out volume ; 23/05/2024
                                <1>     ; cl = left channel volume level (0 to 31)
                                <1>     ; ch = right channel volume level (0 to 31)
                                <1>
00012566 80FB80                 <1>     cmp     bl, 80h
00012569 720A                   <1>     jb      short snd_vol_1
                                <1>     ;ja     snd_nothing ; temporary.
                                <1>     ; 30/07/2022
0001256B 7707                   <1>     ja      short snd_vol_0
                                <1>     ; Set volume level for next play (BL>= 80h)
0001256D 66890D[EC880100]       <1>     mov     [audio_master_volume], cx
```

```
                            <1> snd_vol_0:
00012574 C3                 <1>     retn
                            <1> snd_vol_1:
                            <1>     ; set volume level immediate (BL< 80h)
                            <1>     ;cmp    bl, 0
                            <1>     ;;ja    snd_nothing ; temporary.
                            <1>     ; 30/07/2022
                            <1>     ;ja     short snd_vol_0
                            <1>     ; 23/05/2024
00012575 80FB01             <1>     cmp     bl, 1
00012578 77FA               <1>     ja      short snd_vol_0 ; temporary
                            <1>
0001257A E814020000         <1>     call    snd_dev_check
                            <1>     ;jc     snd_nothing ; temporary.
                            <1>     ; 30/07/2022
0001257F 72F3               <1>     jc      short snd_vol_0
00012581 E81A020000         <1>     call    snd_buf_check
                            <1>     ;jc     snd_nothing ; temporary.
                            <1>     ; 30/07/2022
00012586 72EC               <1>     jc      short snd_vol_0
                            <1>
00012588 A0[B1880100]       <1>     mov     al, [audio_device]
0001258D 3C03               <1>     cmp     al, 3 ; VIA VT 8237R (vt8233)
                            <1>     ;je     vt8233_volume
                            <1>     ; 30/07/2022
0001258F 7207               <1>     jb      short snd_vol_2
                            <1>     ; 28/05/2017
                            <1>     ;ja     snd_nothing ; temporary.
                            <1>     ; 30/07/2022
00012591 77E1               <1>     ja      short snd_vol_0
                            <1>     ;ja     hda_volume
                            <1>     ; 30/07/2022
00012593 E986180000         <1>     jmp     vt8233_volume
                            <1> snd_vol_2:
                            <1>     ; Sound Blaster 16
00012598 3C01               <1>     cmp     al, 1 ; SB 16
                            <1>     ;je     sb16_volume
                            <1>     ; 30/07/2022
0001259A 7705               <1>     ja      short snd_vol_3
0001259C E9021B0000         <1>     jmp     sb16_volume
                            <1> snd_vol_3:
                            <1>     ; 30/07/2022
                            <1>     ;cmp    al, 2
                            <1>     ;ja     short snd_vol_4
000125A1 E9551E0000         <1>     jmp     ac97_volume
                            <1> ;snd_vol_4:
                            <1>     ;jmp    hda_volume
                            <1>
                            <1> soundc_disable:
                            <1>     ; FUNCTION = 12
                            <1>     ; Disable audio device (and unlink DMA memory)
                            <1>     ; 23/08/2024
                            <1>     ; 04/06/2024
                            <1>     ; 30/07/2022
                            <1>     ; 28/05/2017
                            <1>     ; 24/05/2017
                            <1>     ; 22/04/2017
000125A6 E8E8010000         <1>     call    snd_dev_check
                            <1>     ;jc     soundc_dev_err ; temporary.
                            <1>     ; 30/07/2022
000125AB 7305               <1>     jnc     short snd_disable_4
000125AD E962FBFFFF         <1>     jmp     soundc_dev_err
                            <1> snd_disable_4:
                            <1>     ;call   snd_buf_check
                            <1>     ;;jc    sndc_owner_error ; temporary.
                            <1>     ; 30/07/2022
                            <1>     ;jnc    short snd_disable_5
                            <1>     ;jmp    sndc_owner_error
                            <1> ;snd_disable_5:
000125B2 A0[B1880100]       <1>     mov     al, [audio_device]
000125B7 3C03               <1>     cmp     al, 3 ; VIA VT 8237R (vt8233)
000125B9 7414               <1>     je      short snd_disable_1
                            <1>     ;ja     snd_nothing ; temporary.
                            <1>     ; 30/07/2022
000125BB 7766               <1>     ja      short snd_disable_3 ; retn
000125BD 3C01               <1>     cmp     al, 1 ; Sound Blaster 16
000125BF 7507               <1>     jne     short snd_disable_0
000125C1 E8521B0000         <1>     call    sb16_stop
000125C6 EB0C               <1>     jmp     short snd_disable_2
                            <1> snd_disable_0:
000125C8 E8981E0000         <1>     call    ac97_stop
000125CD EB05               <1>     jmp     short snd_disable_2
                            <1> snd_disable_1:
000125CF E833170000         <1>     call    vt8233_stop
                            <1> snd_disable_2:
000125D4 A0[B2880100]       <1>     mov     al, [audio_intr]
                            <1>
                            <1>     ; 15/01/2025
000125D9 20C0               <1>     and     al, al
000125DB 740E               <1>     jz      short snd_disable_5
                            <1>
000125DD 29DB               <1>     sub     ebx, ebx ; 0 = reset
000125DF E840FAFFFF         <1>     call    set_dev_IRQ_service
                            <1>
                            <1>     ;mov    al, [audio_intr]
000125E4 28E4               <1>     sub     ah, ah ; 0 = reset
000125E6 E800F8FFFF         <1>     call    set_hardware_int_vector
                            <1>
                            <1> snd_disable_5:     ; 15/01/2025
000125EB 31C0               <1>     xor     eax, eax
000125ED A2[B1880100]       <1>     mov     byte [audio_device], al
000125F2 A2[B2880100]       <1>     mov     byte [audio_intr], al
000125F7 8705[D0880100]     <1>     xchg    eax, [audio_dma_buff]
                            <1>
                            <1>     ; 15/01/2025
000125FD 09C0               <1>     or      eax, eax
000125FF 7422               <1>     jz      short snd_disable_3
                            <1>
                            <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
00012601 803D[B0880100]00   <1>     cmp     byte [audio_pci], 0 ; AC97 audio controller ?
00012608 7619               <1>     jna     short snd_disable_3
0001260A C605[B0880100]00   <1>     mov     byte [audio_pci], 0
                            <1>
                            <1>     ; 23/08/2024 - bugfix
00012611 3D00000500         <1>     cmp     eax, sb16_dma_buffer ; reserved buffer ?
00012616 740B               <1>     je      short snd_disable_3 ; it isn't an allocated mem buff
                            <1>
                            <1>     ;sub    ecx, ecx
                            <1>     ;xchg   ecx, [audio_dmabuff_size]
00012618 8B0D[D4880100]     <1>     mov     ecx, [audio_dmabuff_size]
```

```
                                <1>     ; 26/11/2023
                                <1>     ; round up (always -rounded up- page count is allocated)
                                <1>     ; ((so deallocation must be done for the rounded up value))
                                <1>     ;add   ecx, PAGE_SIZE - 1   ; 4095
                                <1>     ;call  deallocate_memory_block
                                <1>     ; 04/06/2024
                                <1>     ;call  deallocate_memory_block_x
                                <1>                     ; deallocate ((ecx+4095)>>12) pages
0001261E E83039FFFF             <1>     call   deallocate_memory_block
                                <1> snd_disable_3:
00012623 C3                     <1>     retn
                                <1>
                                <1> sound_dma_map:
                                <1>     ; FUNCTION = 13
                                <1>     ; Map audio dma buff addr to user's buffer addr
                                <1>     ; 30/07/2022
                                <1>     ; 12/05/2017
00012624 21C9                   <1>     and    ecx, ecx
                                <1>     ;jz    sound_buff_error
                                <1>     ; 30/07/2022
00012626 7505                   <1>     jnz    short snd_dma_map_3
00012628 E913FBFFFF             <1>     jmp    sound_buff_error
                                <1> snd_dma_map_3:
0001262D 803D[B1880100]01       <1>     cmp    byte [audio_device], 1
00012634 722A                   <1>     jb     short snd_dma_map_1
                                <1> snd_dma_map_0:
00012636 A1[D0880100]           <1>     mov    eax, [audio_dma_buff]
0001263B 21C0                   <1>     and    eax, eax
0001263D 7421                   <1>     jz     short snd_dma_map_1
                                <1>     ;
0001263F 8A1D[DD880100]         <1>     mov    bl, [audio_user]
00012645 08DB                   <1>     or     bl, bl
00012647 7417                   <1>     jz     short snd_dma_map_1
00012649 3A1D[858E0100]         <1>     cmp    bl, [u.uno]
                                <1>     ;jne   sndc_owner_error
                                <1>     ; 30/07/2022
0001264F 7405                   <1>     je     short snd_dma_map_4
00012651 E910FCFFFF             <1>     jmp    sndc_owner_error
                                <1> snd_dma_map_4:
00012656 8B1D[D4880100]         <1>     mov    ebx, [audio_dmabuff_size]
0001265C 21DB                   <1>     and    ebx, ebx
0001265E 750A                   <1>     jnz    short snd_dma_map_2
                                <1> snd_dma_map_1:
00012660 B800000500             <1>     mov    eax, sb16_dma_buffer
00012665 BB00000100             <1>     mov    ebx, 65536
                                <1> snd_dma_map_2:
0001266A 81C1FF0F0000           <1>     add    ecx, PAGE_SIZE-1 ; 4095
00012670 6681E100F0             <1>     and    cx, ~PAGE_OFF ; not 4095
00012675 39D9                   <1>     cmp    ecx, ebx
                                <1>     ;ja    sound_buff_error
                                <1>     ; 30/07/2022
00012677 7605                   <1>     jna    short snd_dma_map_6
                                <1> snd_dma_map_5:
00012679 E9C2FAFFFF             <1>     jmp    sound_buff_error
                                <1> snd_dma_map_6:
0001267E 50                     <1>     push   eax
0001267F 89D3                   <1>     mov    ebx, edx
00012681 C1E90C                 <1>     shr    ecx, 12 ; byte count to page count
                                <1>     ; eax = physical address of (audio) dma buffer
                                <1>     ; ebx = virtual address of (audio) dma buffer (user's pgdir)
                                <1>     ; ecx = page count (>0)
00012684 E83A39FFFF             <1>     call   direct_memory_access
00012689 58                     <1>     pop    eax
                                <1>     ;jc    sound_buff_error
                                <1>     ; 30/07/2022
0001268A 72ED                   <1>     jc     short snd_dma_map_5
0001268C A3[348E0100]           <1>     mov    [u.r0], eax
00012691 C3                     <1>     retn
                                <1>
                                <1> soundc_info:
                                <1>     ; FUNCTION = 14
                                <1>     ; Get Audio Controller Info
                                <1>     ; 19/11/2023
                                <1>     ; 30/07/2022
                                <1>     ; 10/06/2017
                                <1>     ; 05/06/2017
                                <1>
                                <1>     ;and   bl, bl ; 0
                                <1>     ;jz    short sndc_info_0
                                <1>
                                <1>     ; 19/11/2023
00012692 80FB01                 <1>     cmp    bl, 1
00012695 7609                   <1>     jna    short sndc_info_0
                                <1>
                                <1> ; invalid parameter !
                                <1> ; 30/07/2022
                                <1>     ;mov   eax, ERR_INV_PARAMETER ; 23
                                <1> ;sndc_inf_error:
                                <1> ;   mov    [u.r0], eax
                                <1> ;   mov    [u.error], eax
                                <1> ;   jmp    error
                                <1>     ; 30/07/2022
00012697 29C0                   <1>     sub    eax, eax
00012699 B017                   <1>     mov    al, ERR_INV_PARAMETER ; 23
0001269B E980FAFFFF             <1>     jmp    sysaudio_err
                                <1>
                                <1> sndc_info_0:
000126A0 E8EE000000             <1>     call   snd_dev_check
                                <1>     ;jc    soundc_dev_err
                                <1>     ; 30/07/2022
000126A5 7305                   <1>     jnc    short sndc_info_3
                                <1> snd_data_dev_err:
000126A7 E968FAFFFF             <1>     jmp    soundc_dev_err
                                <1> sndc_info_3:
                                <1>     ; 19/11/2023
                                <1>     ;cmp   bl, 1
                                <1>     ;je    short sndc_info_4
000126AC 20DB                   <1>     and    bl, bl
000126AE 7546                   <1>     jnz    short sndc_info_4
                                <1>
000126B0 8B1D[BC880100]         <1>     mov    ebx, [audio_vendor]
000126B6 8B0D[B8880100]         <1>     mov    ecx, [audio_dev_id]
                                <1>     ;mov   al,  [audio_device]
000126BC 3C02                   <1>     cmp    al, 2 ; AC'97 (ICH)
000126BE 7513                   <1>     jne    short sndc_info_1
                                <1>     ; Intel AC97 (ICH) Audio Controller (=2)
000126C0 668B15[B6880100]       <1>     mov    dx, [NABMBAR]
000126C7 C1E210                 <1>     shl    edx, 16
000126CA 668B15[B4880100]       <1>     mov    dx, [NAMBAR]
000126D1 EB07                   <1>     jmp    short sndc_info_2
                                <1> sndc_info_1:
                                <1>     ; 05/06/2017
                                <1>     ; Note: Intel HDA code (here) is not ready yet!
                                <1>     ; !!! SB16 or VT8233 (VT8237R) !!!
```

```
3396 000126D3 0FB715[B6880100]    <1>     movzx   edx, word [audio_io_base]
3397                              <1> sndc_info_2:
3398 000126DA 88C4                <1>     mov     ah, al ; [audio_device]
3399 000126DC A0[B2880100]        <1>     mov     al, [audio_intr]
3400                              <1>
3401                              <1>     ; EAX = IRQ Number in AL
3402                              <1>     ;       Audio Device Number in AH
3403                              <1>     ; EBX = DEV/VENDOR ID
3404                              <1>     ;       (DDDDDDDDDDDDDDDDVVVVVVVVVVVVVVVV)
3405                              <1>     ; ECX = BUS/DEV/FN
3406                              <1>     ;       (00000000BBBBBBBBDDDDDFFF00000000)
3407                              <1>     ; EDX = NABMBAR/NAMBAR (for AC97)
3408                              <1>     ;       (Low word, DX = NAMBAR address)
3409                              <1>     ; EDX = Base IO Addr (DX) for SB16 & VT8233
3410                              <1>
3411                              <1>     ; 10/06/2017
3412 000126E1 A3[348E0100]        <1>     mov     [u.r0], eax
3413 000126E6 8B2D[308E0100]      <1>     mov     ebp, [u.usp]
3414 000126EC 895D10              <1>     mov     [ebp+16], ebx  ; ebx
3415 000126EF 895514              <1>     mov     [ebp+20], edx  ; edx
3416 000126F2 894D18              <1>     mov     [ebp+24], ecx  ; ecx
3417                              <1>
3418 000126F5 C3                  <1>     retn
3419                              <1>
3420                              <1> sndc_info_4:
3421                              <1>     ; 19/11/2023
3422 000126F6 3C02                <1>     cmp     al, 2 ; Intel AC97 (ICH) Audio Controller
3423 000126F8 7413                <1>     je      short sndc_info_7
3424                              <1> sndc_info_5:
3425                              <1>     ; return ZERO if it is not AC97 audio controller codec
3426 000126FA 31C0                <1>     xor     eax, eax ; 0
3427 000126FC 31DB                <1>     xor     ebx, ebx ; 0
3428                              <1> sndc_info_6:
3429 000126FE A3[348E0100]        <1>     mov     [u.r0], eax
3430 00012703 8B2D[308E0100]      <1>     mov     ebp, [u.usp]
3431 00012709 895D10              <1>     mov     [ebp+16], ebx  ; ebx
3432 0001270C C3                  <1>     retn
3433                              <1> sndc_info_7:
3434 0001270D E8F11F0000          <1>     call    ac97_codec_info
3435 00012712 72E6                <1>     jc      short sndc_info_5
3436                              <1>
3437                              <1>     ; 26/11/2023 - temporary
3438                              <1>     ;and    al, 0FEh       ; clear VRA support bit for test
3439                              <1>
3440 00012714 EBE8                <1>     jmp     short sndc_info_6
3441                              <1>
3442                              <1> sound_data:
3443                              <1>     ; FUNCTION = 15
3444                              <1>     ; Get Current Sound data for graphics
3445                              <1>     ; 30/07/2022
3446                              <1>     ; 22/06/2017
3447                              <1>     ;
3448 00012716 E878000000          <1>     call    snd_dev_check
3449                              <1>     ;jc     soundc_dev_err ; Device not ready !
3450                              <1>     ; 30/07/2022
3451 0001271B 728A                <1>     jc      short snd_data_dev_err
3452                              <1>
3453 0001271D 80FB00              <1>     cmp     bl, 0
3454 00012720 760A                <1>     jna     short sound_data_0
3455                              <1>
3456                              <1>     ; Only PCM OUT buffer data is valid for now!
3457 00012722 B817000000          <1>     mov     eax, ERR_INV_PARAMETER  ; 23
3458 00012727 E9F4F9FFFF          <1>     jmp     sysaudio_err
3459                              <1>
3460                              <1> sound_data_0:
3461 0001272C A1[D0880100]        <1>     mov     eax, [audio_dma_buff]
3462 00012731 09C0                <1>     or      eax, eax
3463                              <1>     ;jz     sound_buff_error
3464                              <1>     ; 30/07/2022
3465 00012733 741A                <1>     jz      short sound_data_5
3466                              <1>
3467 00012735 803D[B1880100]04    <1>     cmp     byte [audio_device], 4 ; Intel HDA
3468 0001273C 744C                <1>     je      short sound_data_4 ; temporary ! (22/06/2017)
3469                              <1>
3470 0001273E 21C9                <1>     and     ecx, ecx
3471                              <1>     ;jnz    short sound_data_1 ; sample transfer
3472                              <1>
3473                              <1>     ; Return only DMA Buffer pointer/offset...
3474                              <1>     ; (If DMA Buffer has been mapped to user's
3475                              <1>     ;  memory space; program can get graphics
3476                              <1>     ;  data by using only this pointer value.)
3477                              <1>
3478                              <1>     ;call   get_dma_buffer_offset
3479                              <1>     ;; eax = DMA buffer offset
3480                              <1>     ;;      (!not half buffer offset!)
3481                              <1>     ;mov    [u.r0], eax
3482                              <1>     ;retn
3483                              <1>
3484                              <1>     ;jz     get_dma_buffer_offset
3485                              <1>     ; 30/07/2022
3486 00012740 7505                <1>     jnz     short sound_data_1
3487 00012742 E95D1F0000          <1>     jmp     get_dma_buffer_offset
3488                              <1>
3489                              <1> sound_data_1:
3490                              <1>     ;mov    eax, [audio_dmabuff_size]
3491                              <1>     ;shr    eax, 1 ; half buffer size
3492                              <1>     ;cmp    ecx, eax
3493                              <1>     ;ja     short sound_buff_error
3494                              <1>
3495 00012747 3B0D[D4880100]      <1>     cmp     ecx, [audio_dmabuff_size]
3496                              <1>     ;ja     sound_buff_error
3497                              <1>     ; 30/07/2022
3498 0001274D 7605                <1>     jna     short sound_data_6
3499                              <1> sound_data_5:
3500 0001274F E9ECF9FFFF          <1>     jmp     sound_buff_error
3501                              <1> sound_data_6:
3502 00012754 89D0                <1>     mov     eax, edx
3503 00012756 25FF0F0000          <1>     and     eax, PAGE_OFF ; 4095 (0FFFh)
3504 0001275B 81F900100000        <1>     cmp     ecx, 4096
3505 00012761 7604                <1>     jna     short sound_data_2
3506                              <1>     ;mov    ecx, 4096 ; max. 1 page
3507                              <1>     ; 30/07/2022
3508 00012763 31C9                <1>     xor     ecx, ecx
3509 00012765 B510                <1>     mov     ch, 16
3510                              <1>     ; ecx = 4096
3511                              <1> sound_data_2:
3512 00012767 01C8                <1>     add     eax, ecx
3513 00012769 3D00100000          <1>     cmp     eax, 4096
3514 0001276E 7606                <1>     jna     short sound_data_3
3515 00012770 6625FF0F            <1>     and     ax, PAGE_OFF ; 4095 (0FFFh)
3516 00012774 29C1                <1>     sub     ecx, eax
3517                              <1>     ; here, ECX has been adjusted to fit
3518                              <1>     ;       in page border..  (<= 4096, >0)
3519                              <1> sound_data_3:
```

```
3520 00012776 51                <1>     push    ecx
3521 00012777 52                <1>     push    edx
3522 00012778 89D3              <1>     mov     ebx, edx
3523 0001277A E88534FFFF        <1>     call    get_physical_addr
3524 0001277F 5A                <1>     pop     edx
3525 00012780 59                <1>     pop     ecx
3526                            <1>     ;jc     sound_buff_error
3527                            <1>     ; 30/07/2022
3528 00012781 72CC              <1>     jc      short sound_data_5
3529                            <1>
3530                            <1>     ; eax = physical address of user's buffer
3531 00012783 89C3              <1>     mov     ebx, eax
3532                            <1>     ; ecx = byte (transfer) count
3533                            <1>     ;call  get_current_sound_data
3534                            <1>     ;retn
3535 00012785 E96F1E0000        <1>     jmp     get_current_sound_data
3536                            <1>
3537                            <1> sound_data_4:
3538                            <1>     ; Intel HDA code is not ready yet !
3539                            <1>     ; 22/06/2017
3540 0001278A 31C0              <1>     xor     eax, eax
3541 0001278C 48                <1>     dec     eax
3542 0001278D A3[348E0100]      <1>     mov     [u.r0], eax ; 0FFFFFFFFh
3543 00012792 C3                <1>     retn
3544                            <1>
3545                            <1> snd_dev_check:
3546                            <1>     ; 10/06/2017
3547                            <1>     ; 05/06/2017
3548                            <1>     ; 24/05/2017
3549                            <1>     ; 22/04/2017
3550                            <1>     ; 21/04/2017
3551                            <1>     ; ... device check at first
3552 00012793 A0[B1880100]      <1>     mov     al, [audio_device]
3553 00012798 3C01              <1>     cmp     al, 1 ; SB 16
3554 0001279A 7203              <1>     jb      short snd_dev_chk_retn ; error !
3555                            <1>     ;cmp    al, 4 ; Intel HDA
3556                            <1>     ;ja     short snd_dbchk_stc ; invalid !
3557                            <1>     ; 10/06/2017
3558 0001279C 3C05              <1>     cmp     al, 5
3559 0001279E F5                <1>     cmc
3560                            <1> snd_dev_chk_retn:
3561 0001279F C3                <1>     retn
3562                            <1>
3563                            <1> snd_buf_check:
3564                            <1>     ; 10/06/2017
3565                            <1>     ; 22/04/2017
3566                            <1>     ; 21/04/2017
3567                            <1>     ; ... buffer & (buffer) owner check at second
3568 000127A0 833D[C4880100]00  <1>     cmp     dword [audio_buffer], 0
3569 000127A7 760D              <1>     jna     short snd_dbchk_stc
3570                            <1> snd_user_check:
3571 000127A9 A0[858E0100]      <1>     mov     al, [u.uno]
3572 000127AE 3A05[DD880100]    <1>     cmp     al, [audio_user]
3573                            <1>     ;jne    short snd_dbchk_stc
3574                            <1>     ;retn
3575 000127B4 74E9              <1>     je      short snd_dev_chk_retn
3576                            <1>
3577                            <1> snd_dbchk_stc:
3578 000127B6 F9                <1>     stc
3579 000127B7 C3                <1>     retn
3580                            <1>
3581                            <1> sound_update:
3582                            <1>     ; FUNCTION = 16
3583                            <1>     ; bl =
3584                            <1>     ;    0 = automatic (sequental) update (with flag switch!)
3585                            <1>     ;    1 = update dma half buffer 1 (without flag switch!)
3586                            <1>     ;    2 = update dma half buffer 2 (without flag switch!)
3587                            <1>     ;  FFh = get current flag value
3588                            <1>     ;       0 = dma half buffer 1 (will be played next)
3589                            <1>     ;       1 = dma half buffer 2 (will be played next)
3590                            <1>     ;
3591                            <1>     ; 29/12/2024
3592                            <1>     ; Note: Requested half buffer is updated and
3593                            <1>     ;       flag is set to other value for other half buffer
3594                            <1>     ;
3595                            <1>     ; For example:
3596                            <1>     ;     * if BL input is 1, half buffer 1 is loaded and
3597                            <1>     ;         flag is switched to 1 -half buff 2 is in order-
3598                            <1>     ;     (next time half buffer 2 is updated automaticly)
3599                            <1>     ;     * if BL input is 2, half buffer 2 is updated/loaded
3600                            <1>     ;         and flag is switched to 0
3601                            <1>     ;
3602                            <1>     ; Return value for BL input = 0,1,2
3603                            <1>     ;     Current flag value (just after switched)
3604                            <1>     ;       -flag will have same value-
3605                            <1>     ;     If BL input is 1, return value will be 1 (1 xor 0)
3606                            <1>     ;     IF BL input is 2, return value will be 0 (1 xor 1)
3607                            <1>
3608                            <1>     ; 28/12/2024
3609                            <1>     ; 05/06/2024
3610                            <1>     ; 30/07/2022
3611                            <1>     ; 10/10/2017
3612                            <1>     ; ... device check at first
3613 000127B8 A0[B1880100]      <1>     mov     al, [audio_device]
3614 000127BD 08C0              <1>     or      al, al ; 0 ; pc speaker or invalid
3615                            <1>     ;jz     soundc_dev_err
3616                            <1>     ; 30/07/2022
3617 000127BF 7505              <1>     jnz     short snd_update_4
3618 000127C1 E94EF9FFFF        <1>     jmp     soundc_dev_err
3619                            <1> snd_update_4:
3620                            <1>     ; ... buffer & (buffer) owner check at second
3621 000127C6 833D[C4880100]00  <1>     cmp     dword [audio_buffer], 0
3622                            <1>     ;jna    sound_buff_error
3623                            <1>     ; 30/07/2022
3624 000127CD 761C              <1>     jna     short snd_update_6 ; jmp sound_buff_error
3625 000127CF A0[858E0100]      <1>     mov     al, [u.uno]
3626 000127D4 3A05[DD880100]    <1>     cmp     al, [audio_user]
3627                            <1>     ;jne    sndc_owner_error
3628                            <1>     ; 30/07/2022
3629 000127DA 7405              <1>     je      short snd_update_5
3630 000127DC E985FAFFFF        <1>     jmp     sndc_owner_error
3631                            <1> snd_update_5:
3632                            <1>     ; Transfer ring 3 (user's) audio buffer content to dma buffer
3633 000127E1 8B3D[D0880100]    <1>     mov     edi, [audio_dma_buff] ; dma buffer (ring 0)
3634 000127E7 09FF              <1>     or      edi, edi
3635                            <1>     ;jz     sound_buff_error
3636                            <1>     ; 30/07/2022
3637 000127E9 7505              <1>     jnz     short snd_update_7
3638                            <1> snd_update_6:
3639 000127EB E950F9FFFF        <1>     jmp     sound_buff_error
3640                            <1> snd_update_7:
3641 000127F0 8B35[C8880100]    <1>     mov     esi, [audio_p_buffer] ; physical address (ring 3)
3642                            <1>     ; 05/06/2024
3643                            <1>     ;mov    ecx, [audio_buff_size]
```

```
                            <1>     ;
                            <1>     ;;;
                            <1>     ; 04/06/2024
                            <1>     ;mov    al, [audio_device]
                            <1>     ;cmp    al, 1
                            <1>     ;je     short snd_update_8 ; SB16
                            <1>     ;and    cl, ~1 ; word alignment
                            <1>     ;cmp    al, 3
                            <1>     ;je     short snd_update_8 ; VIA VT8233
                            <1>     ;; al = 2 ; AC97
                            <1>     ;and    cl, ~7 ; 8 byte alignment
                            <1> ;snd_update_8:
                            <1>     ;;;
                            <1>
                            <1>     ; 05/06/2024
000127F6 8B0D[D8880100]     <1>     mov     ecx, [dma_hbuff_size] ; DMA half buffer size
                            <1>
                            <1>     ;movzx  eax, byte [audio_flag]
000127FC A0[DC880100]       <1>     mov     al, [audio_flag]
00012801 FEC3               <1>     inc     bl
00012803 742F               <1>     jz      short snd_update_3 ; bl = 0FFh
00012805 FECB               <1>     dec     bl
00012807 7417               <1>     jz      short snd_update_0 ; bl = 0
                            <1>
                            <1>     ; 28/12/2024
00012809 80FB02             <1>     cmp     bl, 2
0001280C 7609               <1>     jna     short snd_update_8
                            <1>     ;cmp    bl, 2
                            <1>     ;je     short snd_update_1 ; dma half buffer 2
                            <1>     ;jb     short snd_update_2 ; dma half buffer 1
                            <1>     ; 28/12/2024
                            <1>     ;jz     short  snd_update_2
                            <1>
                            <1>     ; invalid parameter !
                            <1> ; 30/07/2022
                            <1>     ;mov    eax, ERR_INV_PARAMETER ; 23
                            <1> ;   mov     [u.r0], eax
                            <1> ;   mov     [u.error], eax
                            <1> ;   jmp     error
                            <1>
                            <1> snd_update_err:
                            <1>     ; 30/07/2022
0001280E 29C0               <1>     sub     eax, eax
00012810 B017               <1>     mov     al, ERR_INV_PARAMETER ; 23
00012812 E909F9FFFF         <1>     jmp     sysaudio_err
                            <1>
                            <1>     ; 29/12/2024
                            <1> snd_update_8:
00012817 88D8               <1>     mov     al, bl  ; 1 or 2
00012819 FEC8               <1>     dec     al
0001281B A2[DC880100]       <1>     mov     [audio_flag], al ; 0 or 1
                            <1> snd_update_0:
00012820 8035[DC880100]01   <1>     xor     byte [audio_flag], 1 ; update flag !!!
                            <1> snd_update_9:
00012827 3C01               <1>     cmp     al, 1
00012829 7202               <1>     jb      short snd_update_2 ; dma half buffer 1
                            <1> snd_update_1:
                            <1>     ; dma half buffer 2
0001282B 01CF               <1>     add     edi, ecx
                            <1> snd_update_2:
                            <1>     ;rep    movsb
                            <1>     ; 05/06/2024
                            <1>     ;shr    ecx, 2
                            <1>     ;rep    movsd
0001282D F3A4               <1>     rep     movsb ; SB16, AC97, VT8233
                            <1>     ; 29/12/2024
0001282F A0[DC880100]       <1>     mov     al, [audio_flag]
                            <1> snd_update_3:
00012834 A3[348E0100]       <1>     mov     [u.r0], eax
                            <1>
00012839 C3                 <1>     retn
                            <1>
                            <1> sound_getvol:
                            <1>     ; FUNCTION = 17
                            <1>     ; Get sound volume level
                            <1>     ; 24/05/2024
                            <1>     ; bl = component
                            <1>     ;              0 = master/playback/lineout volume
                            <1>     ;              1 = PCM out volume
                            <1>     ; Return:
                            <1>     ; cl = left channel volume level (0 to 31)
                            <1>     ; ch = right channel volume level (0 to 31)
                            <1>
0001283A 80FB01             <1>     cmp     bl, 1
0001283D 7709               <1>     ja      short snd_gvol_0 ; temporary ; 24/05/2024
0001283F 7408               <1>     je      short snd_gvol_1
00012841 668B0D[EC880100]   <1>     mov     cx, [audio_master_volume]
                            <1> snd_gvol_0:
00012848 C3                 <1>     retn
                            <1> snd_gvol_1:
00012849 668B0D[EE880100]   <1>     mov     cx, [audio_pcmo_volume]
00012850 C3                 <1>     retn
                            <1>
                            <1> %if 0
                            <1>
                            <1> sound_dmaclear:
                            <1>     ; FUNCTION = 18
                            <1>     ; Clear DMA buffer
                            <1>     ;       (may be useful for non-VRA AC97 hardware)
                            <1>     ; bl = cleaning byte value
                            <1>     ;      0 or 80h (not a necessary value, 0 is most proper)
                            <1>     ;
                            <1>     ; Return value
                            <1>     ;      eax = DMA half buffer size (DMA buff size / 2)
                            <1>     ;  If eax = 0
                            <1>     ;      there is not an active DMA buffer for user)
                            <1>     ;
                            <1>     ; 28/12/2024
                            <1>     mov     edi, [audio_dma_buff]
                            <1>     and     edi, edi
                            <1>     jz      short snd_dma_clear_ok
                            <1>     mov     ecx, [audio_dmabuff_size]
                            <1>     or      ecx, ecx
                            <1>     jz      short snd_dma_clear_ok
                            <1>     ; security (is necessary for multitasking)
                            <1>     mov     cl, [audio_user]
                            <1>     or      cl, cl
                            <1>     jz      short snd_dma_clear_ok
                            <1>     cmp     cl, [u.uno]
                            <1>     jne     short snd_dma_clear_ok
                            <1>     mov     ecx, [dma_hbuff_size]
                            <1>     and     ecx, ecx
                            <1>     jz      short snd_dma_clear_ok
                            <1>
```

```
3768                                <1>     mov     [u.r0], ecx
3769                                <1>     shl     ecx, 1 ; full buffer size in bytes
3770                                <1>     mov     al, bl
3771                                <1>     rep     stosb
3772                                <1>
3773                                <1> snd_dma_clear_ok:
3774                                <1>     retn
3775                                <1>
3776                                <1> %endif
3777                                <1>
3778                                <1> set_irq_callback_service:
3779                                <1>     ; 23/11/2023
3780                                <1>     ; 20/11/2023 (TRDOS 386 Kernel v2.0.7)
3781                                <1>     ; 30/07/2022 (TRDOS 386 Kernel v2.0.5)
3782                                <1>     ; 03/08/2020
3783                                <1>     ; 10/06/2017
3784                                <1>     ; 12/05/2017
3785                                <1>     ; 24/04/2017
3786                                <1>     ; 22/04/2017
3787                                <1>     ; caller: 'syscalbac' or 'sysaudio' or ...
3788                                <1>     ; 13/04/2017, 14/04/2017, 17/04/2017
3789                                <1>     ; 24/02/2017, 26/02/2017, 28/02/2017
3790                                <1>     ; 21/02/2017 - TRDOS 386 (TRDOS v2.0)
3791                                <1>     ;
3792                                <1>     ; Link or unlink IRQ callback service to/from user (ring 3)
3793                                <1>     ;
3794                                <1>     ; INPUT ->
3795                                <1>     ;       If AL = 0, the caller is 'syscalbac';
3796                                <1>     ;          otherwise, the caller is 'sysaudio' or ...
3797                                <1>     ;          (AL = user number)
3798                                <1>     ;
3799                                <1>     ;       BL = IRQ number (Hardware interrupt request number)
3800                                <1>     ;            (0 to 15 but IRQ 0,1,2,6,8,14,15 are prohibited)
3801                                <1>     ;            IRQ numbers 3,4,5,7,9,10,11,12,13 are valid
3802                                <1>     ;            (numbers >15 are invalid)
3803                                <1>     ;
3804                                <1>     ;       BH = 0 = Unlink IRQ (in BL) from user (ring 3) service
3805                                <1>     ;            1 = Link IRQ by using Signal Response Byte method
3806                                <1>     ;            2 = Link IRQ by using Callback service method
3807                                <1>     ;            3 = Link IRQ by using Auto Increment S.R.B. method
3808                                <1>     ;           >3 = invalid
3809                                <1>     ;                (syscallback version will return to user)
3810                                <1>     ;
3811                                <1>     ;       CL = Signal Return/Response Byte value
3812                                <1>     ;
3813                                <1>     ;       If BH = 3, kernel will put a counter value ; 03/08/2020
3814                                <1>     ;                 (into the S.R.B. addr)
3815                                <1>     ;                between 0 to 255. (start value = CL+1)
3816                                <1>     ;
3817                                <1>     ;       NOTE: counter value, for example: even and odd numbers
3818                                <1>     ;             may be used for -audio- DMA buffer switch
3819                                <1>     ;             within double buffer method, etc.
3820                                <1>     ;
3821                                <1>     ;       EDX = Signal return (Response) byte address
3822                                <1>     ;                        - or -
3823                                <1>     ;             Interrupt/Callback service/routine address
3824                                <1>     ;
3825                                <1>     ;             (virtual address in user's memory space)
3826                                <1>     ;
3827                                <1>     ; OUTPUT ->
3828                                <1>     ;       CF = 0 & EAX = 0 -> Successful setting
3829                                <1>     ;       CF = 1 & EAX > 0 -> IRQ is prohibited or locked
3830                                <1>     ;                           by another process
3831                                <1>     ;                  eax = ERR_PERM_DENIED -> prohibited or locked
3832                                <1>     ;                  eax = ERR_INV_PARAMETER ->
3833                                <1>     ;                        invalid parameter/option or bad address
3834                                <1>     ;
3835                                <1>     ; TRDOS 386 - IRQ CALLBACK structures (parameters):
3836                                <1>     ;
3837                                <1>     ;          [u.irqlock] : 1 word, IRQ flags (0-15) that indicates
3838                                <1>     ;                        which IRQs are locked by (that) user.
3839                                <1>     ;                         Lock and unlock (by user) will change
3840                                <1>     ;                        these flags or 'terminate process' (sysexit)
3841                                <1>     ;                        will clear these flags and unlock those IRQs.
3842                                <1>     ;
3843                                <1>     ;                        Bit 0 is for IRQ 0 and Bit 15 is for IRQ 15
3844                                <1>     ;
3845                                <1>     ;          IRQ(x).owner       : 1 byte, user, [u.uno], 0 = free (unlocked)
3846                                <1>     ;
3847                                <1>     ;          IRQ(x).method : 1 byte for callback method & status
3848                                <1>     ;                          0 = Signal Response Byte method
3849                                <1>     ;                          1 = Callback service method
3850                                <1>     ;                          >1 = invalid for current 'syscalback'.
3851                                <1>     ;                        or(+) 80h = IRQ is in use by system (ring 0)
3852                                <1>     ;                                    function (audio etc.) or
3853                                <1>     ;                                   a device driver.
3854                                <1>     ;                        (system function will ignore the lock/owner)
3855                                <1>     ;
3856                                <1>     ;          IRQ(x).srb  : 1 byte, Signal Return/Response byte value
3857                                <1>     ;                         (a fixed value by user or a counter value
3858                                <1>     ;                        from 0 to 255, which is increased by every
3859                                <1>     ;                        interrupt just before putting it into
3860                                <1>     ;                        the Signal Response byte address)
3861                                <1>     ;                        (This is not used in callback serv method)
3862                                <1>     ;
3863                                <1>     ;          IRQ(x).addr : 1 dword
3864                                <1>     ;                         Signal Response Byte address (physical)
3865                                <1>     ;                                     -or-
3866                                <1>     ;                         Callback service address (virtual)
3867                                <1>     ;
3868                                <1>     ;          IRQ(x).dev  : 1 byte
3869                                <1>     ;                         0 = Default device or kernel function
3870                                <1>     ;                                     -or-
3871                                <1>     ;                         1-255 = Assigned device driver number
3872                                <1>     ;
3873                                <1>     ;          (x) = 3,4,5,7,9,10,11,12,13
3874                                <1>     ;
3875                                <1>
3876 00012851 80FB0F               <1>     cmp     bl, 15
3877 00012854 7727                 <1>     ja      short scbs_2
3878                                <1>
3879 00012856 80FF03               <1>     cmp     bh, 3
3880 00012859 7722                 <1>     ja      short scbs_2  ; invalid parameter
3881                                <1>
3882 0001285B 0FB6FB               <1>     movzx   edi, bl ; save IRQ number
3883                                <1>
3884                                <1>             ; IRQ 0,1,2,6,8,14,15 are prohibited
3885                                <1>     ;IRQenum: ; 'trdosk9.s'
3886                                <1>     ;       db  0,0,0,1,2,3,0,4,0,5,6,7,8,9,0,0
3887                                <1>
3888 0001285E 0FB6B7[80350100]     <1>     movzx   esi, byte [edi+IRQenum] ; IRQ availability
3889                                <1>                                     ; enumeration/index
3890                                <1>     ; 20/11/2023
3891 00012865 4E                   <1>     dec     esi
```

```
                                <1>     ;dec    si
00012866 780F                   <1>     js      short scbs_1 ;  0 -> 0FFFFh
                                <1>
                                <1>     ; ESI = IRQ callback parameters index number (0 to 8)
                                <1>
00012868 08FF                   <1>     or      bh, bh
0001286A 7417                   <1>     jz      short scbs_4 ; unlink the IRQ (in BL)
                                <1>
0001286C FECF                   <1>     dec     bh
                                <1>     ; bh = method (0 = signal response byte, 1 = callback)
                                <1>     ;             (2 = auto increment of signal response byte)
                                <1>
0001286E 80BE[60880100]00       <1>     cmp     byte [esi+IRQ.owner], 0 ; locked ?
00012875 7635                   <1>     jna     short scbs_6   ; no... OK...
                                <1>
                                <1> scbs_1:
                                <1>     ; permission denied (prohibited IRQ)
                                <1>     ;mov    eax, ERR_PERM_DENIED
                                <1>     ; 30/07/2022
00012877 29C0                   <1>     sub     eax, eax
00012879 B00B                   <1>     mov     al, ERR_PERM_DENIED
0001287B F9                     <1>     stc
0001287C C3                     <1>     retn
                                <1> scbs_2:
                                <1>     ;stc
                                <1> scbs_3:
                                <1>     ;mov    eax, ERR_INV_PARAMETER
                                <1>     ; 30/07/2022
0001287D 29C0                   <1>     sub     eax, eax
0001287F B017                   <1>     mov     al, ERR_INV_PARAMETER
00012881 F9                     <1>     stc
00012882 C3                     <1>     retn
                                <1>
                                <1> scbs_4: ; unlink the requested IRQ (if it belongs to current user)
                                <1>     ; 10/06/2017
                                <1>     ; 22/04/2017
                                <1>     ; 14/04/2017
                                <1>     ; If AL = 0 -> The caller is 'syscalbac'
00012883 8AA6[60880100]         <1>     mov     ah, [esi+IRQ.owner]
00012889 3A25[858E0100]         <1>     cmp     ah, [u.uno]
0001288F 75E6                   <1>     jne     short scbs_1
                                <1>
00012891 FE0D[AE8E0100]         <1>     dec     byte [u.irqc] ; decrease IRQ count (in use)
                                <1>
                                <1>     ;sub    ah, ah
                                <1>     ;mov    [esi+IRQ.owner], ah ; 0 ; free !!!
                                <1>     ;and    byte [esi+IRQ.method], 80h
                                <1>     ;mov    [esi+IRQ.srb], ah ; 0
                                <1>     ;mov    [esi+IRQ.dev], ah ; 0
                                <1>     ;mov    dword [esi+IRQ.addr], 0
                                <1>     ;mov    dword [u.r0], 0
                                <1>
                                <1>     ;mov    byte [esi+IRQ.owner], 0
                                <1>
                                <1>     ; 22/04/2017
00012897 29C0                   <1>     sub     eax, eax
00012899 8886[60880100]         <1>     mov     [esi+IRQ.owner], al ; 0
                                <1>     ; 10/06/2017
0001289F 8686[72880100]         <1>     xchg    al, [esi+IRQ.method]
000128A5 2480                   <1>     and     al, 80h
000128A7 745D                   <1>     jz      short scbs_12
                                <1>     ; Audio device must be disabled -later- ! ([IRQ.medhod] = 80h)
                                <1>
                                <1> ;   cmp     byte [esi+IRQ.method], 80h ; device drv or kernel extension ?
                                <1> ;   jb      short scbs_12 ; bh = 0 reset to default IRQ handler
                                <1> ;
                                <1> ;   and     al, al
                                <1> ;   jz      short scbs_5  ; the caller is 'syscalbac'
                                <1> ;   ; The caller is 'sysaudio' or ...
000128A9 30C0                   <1>     xor     al, al
                                <1> ;   mov     [esi+IRQ.method], al ; 0 ; reset kernel extension flag
                                <1> ;scbs_5:
                                <1> ;   sub     ah, ah
                                <1>     ;mov    [u.r0], eax ; 0
000128AB C3                     <1>     retn
                                <1>
                                <1> scbs_6:
                                <1>     ; 14/04/2017
000128AC 20C0                   <1>     and     al, al
000128AE 7405                   <1>     jz      short scbs_7  ; the caller is 'syscalbac'
                                <1>     ; AL = user number ([u.uno] or [audio.user] or ...)
                                <1>     ; The caller is 'sysaudio' or ...
                                <1>     ;
                                <1>     ; bh = method (0 = signal response byte, 1 = callback)
                                <1>     ;             (2 = auto increment of signal response byte)
                                <1>
000128B0 80CF80                 <1>     or      bh, 80h        ; Kernel extension flag !
000128B3 EB0A                   <1>     jmp     short scbs_8
                                <1> scbs_7:
000128B5 8A86[72880100]         <1>     mov     al, [esi+IRQ.method] ; >= 80h = kernel is using this IRQ
000128BB 2480                   <1>     and     al, 80h ; use only bit 7 (kernel function flag)
000128BD 08C7                   <1>     or      bh, al          ; method
                                <1>                             ; 0 = signal response byte, 1 = callback
                                <1>                             ; 2 = auto increment of s.r.b.
                                <1> scbs_8:
000128BF A0[858E0100]           <1>     mov     al, [u.uno] ; user (process) number (1 to 16)
000128C4 8886[60880100]         <1>     mov     [esi+IRQ.owner], al ; lock the IRQ for user
000128CA 88BE[72880100]         <1>     mov     [esi+IRQ.method], bh
                                <1>
                                <1> ;   test    bh, 1
                                <1> ;   jnz     short scbs_9    ; Callback method, CX will not be used
                                <1> ;
                                <1> ;   test    bh, 2           ; use auto increment (counter) method
                                <1> ;   jz      short scbs_10   ; (count can be used for buffer switch)
                                <1> ;scbs_9:
                                <1> ;   xor     ecx, ecx ; 0
                                <1>
                                <1> scbs_10:
                                <1>     ;mov    [esi+IRQ.method], bh
000128D0 888E[7B880100]         <1>     mov     [esi+IRQ.srb], cl
000128D6 C686[69880100]00       <1>     mov     byte [esi+IRQ.dev], 0 ; device number is always 0
                                <1>                             ; for this system call
                                <1>     ;test   bh, 1
000128DD 80E701                 <1>     and     bh, 1 ; 17/04/2017
000128E0 7513                   <1>     jnz     short scbs_11   ; callback method, use virtual address
                                <1>
000128E2 53                     <1>     push    ebx ; IRQ number (in BL)
000128E3 89D3                   <1>     mov     ebx, edx
                                <1>     ; ebx = virtual address
                                <1>     ; [u.pgdir] = page directory's physical address
000128E5 FE05[1E880100]         <1>     inc     byte [no_page_swap] ; 1
                                <1>                     ; Do not add this page to swap queue
                                <1>                     ; and remove it from swap queue if it is
                                <1>                     ; on the queue.
```

```
4016 000128EB E81433FFFF          <1>     call    get_physical_addr
4017 000128F0 5B                  <1>     pop     ebx
4018 000128F1 728A                <1>     jc      short scbs_3 ; invalid address !
4019                              <1>     ; eax = physical address of the virtual address in user's space
4020 000128F3 89C2                <1>     mov     edx, eax
4021                              <1> scbs_11:
4022                              <1>     ;shl    si, 2           ; byte (index) to dword (offset)
4023                              <1>     ; 23/11/2023
4024 000128F5 C1E602              <1>     shl     esi, 2
4025 000128F8 8996[84880100]      <1>     mov     [esi+IRQ.addr], edx
4026                              <1>
4027 000128FE FE05[AE8E0100]      <1>     inc     byte [u.irqc]  ; increase IRQ (in use) count
4028                              <1>
4029 00012904 FEC7                <1>     inc     bh ; 17/04/2017
4030                              <1>     ; bh > 0 -> set to requested IRQ handler (IRQ_u_list)
4031                              <1> scbs_12:
4032 00012906 88D8                <1>     mov     al, bl ; IRQ number
4033 00012908 88FC                <1>     mov     ah, bh ; 0 = reset, >0 = set
4034 0001290A E8DCF4FFFF          <1>     call    set_hardware_int_vector
4035                              <1>
4036 0001290F 31C0                <1>     xor     eax, eax
4037                              <1>     ;mov    [u.r0], eax ; 0
4038                              <1>
4039 00012911 C3                  <1>     retn    ; return with success (cf=0, eax=0)
4040                              <1>
4041                              <1> sysdma: ; DMA FUNCTIONS
4042                              <1>     ; 08/08/2022
4043                              <1>     ; 30/07/2022 - TRDOS 386 Kernel v2.0.5
4044                              <1>     ; 02/09/2017
4045                              <1>     ; 28/08/2017
4046                              <1>     ; 20/08/2017 - TRDOS 386 (TRDOS v2.0)
4047                              <1>     ;
4048                              <1>     ; Inputs:
4049                              <1>     ;       BH = 0 -> Allocate DMA buffer
4050                              <1>     ;          BL = 0 -> Use the system's default DMA
4051                              <1>     ;                    (SB16) Buffer
4052                              <1>     ;               Buffer Size (max.) = 65536 bytes
4053                              <1>     ;          BL > 0 -> Allocate (a new) DMA buffer
4054                              <1>     ;          ECX = DMA Buffer Size in bytes (<=128KB)
4055                              <1>     ;          EDX = Virtual Address of DMA buffer
4056                              <1>     ;
4057                              <1>     ;       BH = 1 -> Initialize (Start) DMA service
4058                              <1>     ;            BL, bit 0 to 3 = Channel Number (0 to 7)
4059                              <1>     ;            BL, bit 7 = Auto Initialized Mode
4060                              <1>     ;                        (If bit 7 is set)
4061                              <1>     ;                   bit 6 = Record (read) mode (0= playback)
4062                              <1>     ;            ECX = byte count (0 = use dma buffer size)
4063                              <1>     ;            EDX = physical buffer address
4064                              <1>     ;                  (0 = use dma buffer -start- address)
4065                              <1>     ;
4066                              <1>     ;       BH = 2 -> Get Current DMA Buffer Offset
4067                              <1>     ;            BL = DMA channel number
4068                              <1>     ;
4069                              <1>     ;       BH = 3 -> Get Current DMA count down value
4070                              <1>     ;            BL = DMA channel number (0 tO 7)
4071                              <1>     ;
4072                              <1>     ;       BH = 4 -> Get Current DMA channel (in progress)
4073                              <1>     ;
4074                              <1>     ;       BH = 5 -> Get System's Default DMA Buffer Address
4075                              <1>     ;
4076                              <1>     ;       BH = 6 -> Get Current DMA Buffer Address
4077                              <1>     ;
4078                              <1>     ;       BH = 7 -> Stop DMA service
4079                              <1>     ;
4080                              <1>     ; Outputs:
4081                              <1>     ;
4082                              <1>     ;       For BH = 0 ; Allocate DMA buffer
4083                              <1>     ;           EAX = Physical address of DMA buffer
4084                              <1>     ;           ECX = Allocated buffer size in bytes
4085                              <1>     ;                 - page count * 4096 -
4086                              <1>     ;                (may be bigger than requested)
4087                              <1>     ;           If BL input > 0,
4088                              <1>     ;              'sysalloc:' system call will be used with
4089                              <1>     ;              EBX (for 'sysalloc') = EDX (for 'sysdma')
4090                              <1>     ;              ECX is same, byte count (buffer size)
4091                              <1>     ;              EDX = 1024*1024*16 ; 16 MB upper limit
4092                              <1>     ;           If BL input = 0,
4093                              <1>     ;              Default DMA buffer (SB16 buffer) will be
4094                              <1>     ;              checked and if it is free, it's address
4095                              <1>     ;              will be returned in EAX and it's size
4096                              <1>     ;              will be returned in ECX (as 65536)
4097                              <1>     ;
4098                              <1>     ;           If CF = 1, error code is in EAX
4099                              <1>     ;              EAX = -1 ; DMA buffer allocation error!
4100                              <1>     ;              EAX = 11 ; 'Permission Denied' error !
4101                              <1>     ;
4102                              <1>     ;              Note: 'sysalloc' error return method
4103                              <1>     ;                    will be applied if BL input > 0 !
4104                              <1>     ;
4105                              <1>     ;       For BH = 1 ; Initialize (Start) DMA
4106                              <1>     ;           EAX = 0 (Successful)
4107                              <1>     ;           If CF = 1, error code is in EAX
4108                              <1>     ;
4109                              <1>     ;       For BH = 2 ; Get Current DMA Buffer Offset
4110                              <1>     ;           EAX = DMA Buffer Offset (in bytes)
4111                              <1>     ;           ;
4112                              <1>     ;            AX = DMA buffer offset
4113                              <1>     ;           EAX bits 16 to 23 = Page register value
4114                              <1>     ;
4115                              <1>     ;       For BH = 3 ; Get Current DMA count down value
4116                              <1>     ;           EAX = Count down value (remain bytes)
4117                              <1>     ;
4118                              <1>     ;       For BH = 4 ; Get Current DMA channel (in progress)
4119                              <1>     ;           EAX = DMA channel number (0 to 7)
4120                              <1>     ;             AH = 0 if the owner is the caller process
4121                              <1>     ;             AH > 0 if the dma channel is in use by
4122                              <1>     ;                    another user/process
4123                              <1>     ;           EAX = -1 (0FFFFFFFFh)
4124                              <1>     ;                 if DMA service is not in use
4125                              <1>     ;                 (stopped or not initialized/started)
4126                              <1>     ;
4127                              <1>     ;       For BH = 5 ; Get System's Default DMA Buff Addr
4128                              <1>     ;           EAX = Default DMA Buffer Address (Physical)
4129                              <1>     ;              = offset 'sb16_dma_buffer:'
4130                              <1>     ;           ECX = Buffer size
4131                              <1>     ;              = 65536
4132                              <1>     ;
4133                              <1>     ;       For BH = 6 ; Get Current DMA Buffer Address
4134                              <1>     ;           EAX = Current DMA buffer address (Physical)
4135                              <1>     ;           ECX = Current DMA buffer size (setting value)
4136                              <1>     ;           Note: These values are for current dma channel
4137                              <1>     ;                 settings for the user/process
4138                              <1>     ;           ** For now (for current TRDOS 386 version)
4139                              <1>     ;             only one user/process can use only one
```

```
                                <1>     ;             dma channel & one dma buffer at same time
                                <1>     ;             (no multi tasking on DMA service) !!! **
                                <1>     ;            (Once, current DMA user must stop it's own DMA
                                <1>     ;             DMA service, than another user/program
                                <1>     ;             can use DMA service with same dma channel
                                <1>     ;             or with another DMA channel.)
                                <1>     ;
                                <1>     ;       For BH = 7 ; Stop DMA service (for current user
                                <1>     ;           and current DMA channel)
                                <1>     ;           EAX = 0 ; successful
                                <1>     ;           CF = 1 & EAX > 0 (= -1) -> Error
                                <1>
00012912 80FF07                 <1>     cmp     bh, 7
00012915 7612                   <1>     jna     short sysdma_0
                                <1>
                                <1> sysdma_err:
00012917 31C0                   <1>     xor     eax, eax
00012919 48                     <1>     dec     eax ; -1
                                <1> sysdma_perm_err:
0001291A A3[348E0100]           <1>     mov     [u.r0], eax
0001291F A3[A08E0100]           <1>     mov     [u.error], eax ; DMA service error !
00012924 E935A1FFFF             <1>     jmp     error
                                <1>
                                <1> sysdma_0:
00012929 08FF                   <1>     or      bh, bh
0001292B 7537                   <1>     jnz     short sysdma_1 ; 30/07/2022
                                <1>
0001292D 20DB                   <1>     and     bl, bl
0001292F 7416                   <1>     jz      short sysdma_01
                                <1>
                                <1>     ; redirect system call to 'sysalloc'
00012931 89D3                   <1>     mov     ebx, edx ; virtual address of DMA buffer
                                <1>     ;ecx = Buffer size in bytes
                                <1>     ; DMA buffer address <= 16MB upper limit
00012933 BA00000001             <1>     mov     edx, 1024*1024*16 ; 16MB limit for DMA buff
                                <1>
00012938 C705[F88C0100]FFFF-    <1>     mov     dword [dma_addr], 0FFFFFFFFh ; -1
00012940 FFFF                   <1>
                                <1>
00012942 E94DE8FFFF             <1>     jmp     sysalloc
                                <1>
                                <1> sysdma_01:
00012947 B800000500             <1>     mov     eax, sb16_dma_buffer
                                <1>
0001294C 803D[B1880100]01       <1>     cmp     byte [audio_device], 1
00012953 724A                   <1>     jb      short sysdma_03
                                <1>
00012955 3B05[D0880100]         <1>     cmp     eax, [audio_dma_buff]
0001295B 7527                   <1>     jne     short sysdma_02
                                <1>
                                <1> sysdma_0_err:
0001295D B80B000000             <1>     mov     eax, ERR_PERM_DENIED
00012962 EBB6                   <1>     jmp     short sysdma_perm_err
                                <1>
                                <1>     ; 30/07/2022
                                <1> sysdma_1:
00012964 80FF01                 <1>     cmp     bh, 1
                                <1>     ;ja     sysdma_5
                                <1>     ; 30/07/2022
00012967 7605                   <1>     jna     short sysdma_10
00012969 E90C010000             <1>     jmp     sysdma_5
                                <1>
                                <1> sysdma_10:
0001296E F6C340                 <1>     test    bl, 40h ; record (read) mode -BL, bit 6-
                                <1>     ;jnz    sysdma_err ; not ready yet!
                                <1>     ; 30/07/2022
00012971 757C                   <1>     jnz     short sysdma_06 ; jmp sysdma_err
                                <1>
00012973 A1[F88C0100]           <1>     mov     eax, [dma_addr] ; physical address of dma buffer
00012978 21C0                   <1>     and     eax, eax
                                <1>     ;jz     sysdma_err
                                <1>     ; 30/07/2022
0001297A 7473                   <1>     jz      short sysdma_06 ; jmp sysdma_err
                                <1>
0001297C 09D2                   <1>     or      edx, edx
0001297E 7574                   <1>     jnz     short sysdma_11
                                <1>
00012980 89C2                   <1>     mov     edx, eax
00012982 EB74                   <1>     jmp     short sysdma_12
                                <1>
                                <1> sysdma_02:
                                <1>     ; Only one user is permitted for audio/dma functions
                                <1>
00012984 833D[D0880100]00       <1>     cmp     dword [audio_dma_buff], 0
0001298B 7612                   <1>     jna     short sysdma_03
                                <1>
0001298D 8A1D[DD880100]         <1>     mov     bl, [audio_user]
00012993 08DB                   <1>     or      bl, bl
00012995 7408                   <1>     jz      short sysdma_03
                                <1>
00012997 3A1D[858E0100]         <1>     cmp     bl, [u.uno]
0001299D 75BE                   <1>     jne     short sysdma_0_err
                                <1>
                                <1> sysdma_03:
0001299F 8A1D[F58C0100]         <1>     mov     bl, [dma_user]
000129A5 20DB                   <1>     and     bl, bl
000129A7 750E                   <1>     jnz     short sysdma_04
                                <1>
000129A9 8A1D[858E0100]         <1>     mov     bl, [u.uno]
000129AF 881D[F58C0100]         <1>     mov     [dma_user], bl
                                <1>
000129B5 EB15                   <1>     jmp     short sysdma_05
                                <1>
                                <1> sysdma_04:
000129B7 8B35[F88C0100]         <1>     mov     esi, [dma_addr]
000129BD 21F6                   <1>     and     esi, esi
000129BF 740B                   <1>     jz      short sysdma_05
                                <1>
000129C1 46                     <1>     inc     esi ; -1 -> 0
000129C2 7408                   <1>     jz      short sysdma_05
                                <1>
000129C4 3A1D[858E0100]         <1>     cmp     bl, [u.uno]
000129CA 7591                   <1>     jne     short sysdma_0_err
                                <1>
                                <1> sysdma_05:
                                <1>     ; edx = virtual address (user's buffer address)
                                <1>     ;
000129CC 81F900000100           <1>     cmp     ecx, 65536   ; byte count (buffer size)
                                <1>     ;ja     sysdma_err
                                <1>     ; 30/07/2022
000129D2 771B                   <1>     ja      short sysdma_06 ; jmp sysdma_err
                                <1>     ;
000129D4 81C1FF0F0000           <1>     add     ecx, PAGE_SIZE-1 ; 4095
000129DA 6681E100F0             <1>     and     cx, ~PAGE_OFF ; not 4095
```

```
                            <1>     ;cmp    ecx, 65536
                            <1>     ;ja     sysdma_err
000129DF 51                 <1>     push    ecx     ; buffer size (allocated pages * 4096)
000129E0 50                 <1>     push    eax     ; offset sb16_dma_buffer
000129E1 89D3               <1>     mov     ebx, edx
000129E3 C1E90C             <1>     shr     ecx, 12 ; byte count to page count
                            <1>     ; eax = physical address of (audio) dma buffer
                            <1>     ; ebx = virtual address of (audio) dma buffer (user's pgdir)
                            <1>     ; ecx = page count (>0)
000129E6 E8D835FFFF         <1>     call    direct_memory_access
000129EB 58                 <1>     pop     eax
000129EC 59                 <1>     pop     ecx
                            <1>     ;jc     sysdma_err
                            <1>     ; 30/07/2022
000129ED 7315               <1>     jnc     short sysdma_07
                            <1> sysdma_06:
000129EF E923FFFFFF         <1>     jmp     sysdma_err
                            <1>
                            <1> sysdma_11:
000129F4 39C2               <1>     cmp     edx, eax
                            <1>     ;jb     sysdma_err
                            <1>     ; 30/07/2022
000129F6 72F7               <1>     jb      short sysdma_06 ; jmp sysdma_err
                            <1> sysdma_12:
000129F8 21C9               <1>     and     ecx, ecx
000129FA 7515               <1>     jnz     short sysdma_13
                            <1>
000129FC 8B0D[FC8C0100]     <1>     mov     ecx, [dma_size]
00012A02 EB15               <1>     jmp     short sysdma_14
                            <1>
                            <1> sysdma_07:
00012A04 A3[F88C0100]       <1>     mov     [dma_addr], eax
00012A09 890D[FC8C0100]     <1>     mov     [dma_size], ecx ; dma buffer size (in bytes)
                            <1>
                            <1>     ;mov    [u.r0], eax ; DMA Buffer Address (Physical)
                            <1>
                            <1>     ;mov    ebp, [u.usp]  ; ebp points to user's registers
                            <1>     ;mov    [ebp+24], ecx ; return to user with ecx value
                            <1>
                            <1>     ;jmp    sysret
                            <1>
                            <1>     ; 28/08/2017
00012A0F EB7A               <1>     jmp     sysdma_51
                            <1>
                            <1> sysdma_13:
00012A11 3B0D[FC8C0100]     <1>     cmp     ecx, [dma_size]
                            <1>     ;ja     sysdma_err
                            <1>     ; 30/07/2022
00012A17 77D6               <1>     ja      short sysdma_06 ; jmp sysdma_err
                            <1> sysdma_14:
00012A19 89C6               <1>     mov     esi, eax
00012A1B 0335[FC8C0100]     <1>     add     esi, [dma_size]
                            <1>
00012A21 89D0               <1>     mov     eax, edx
00012A23 01C8               <1>     add     eax, ecx
                            <1>     ;jc     sysdma_err ; 02/09/2017
                            <1>     ; 30/07/2022
00012A25 72C8               <1>     jc      short sysdma_06 ; jmp sysdma_err
                            <1>
00012A27 39F0               <1>     cmp     eax, esi
                            <1>     ;ja     sysdma_err
                            <1>     ; 30/07/2022
00012A29 77C4               <1>     ja      short sysdma_06 ; jmp sysdma_err
                            <1>
00012A2B 8B3D[D0880100]     <1>     mov     edi, [audio_dma_buff]
00012A31 8B35[F88C0100]     <1>     mov     esi, [dma_addr]
                            <1>
00012A37 09FF               <1>     or      edi, edi
00012A39 7425               <1>     jz      short sysdma_16
                            <1>
00012A3B 803D[B1880100]01   <1>     cmp     byte [audio_device], 1
00012A42 7209               <1>     jb      short sysdma_15
                            <1>
                            <1>     ; Sound Blaster 16
00012A44 39FE               <1>     cmp     esi, edi
                            <1>     ;je     sysdma_0_err ; permmission denied !
                            <1>     ; 30/07/2022
00012A46 7505               <1>     jne     short sysdma_15
00012A48 E910FFFFFF         <1>     jmp     sysdma_0_err
                            <1> sysdma_15:
00012A4D C605[F78C0100]48   <1>     mov     byte [dma_mode], 48h ; single mode playback
                            <1>
00012A54 F6C380             <1>     test    bl, 80h ; DMA mode - BL, bit 7, auto init -
00012A57 7407               <1>     jz      short sysdma_16
                            <1>     ; Auto initialized playback (write) mode
00012A59 8005[F78C0100]10   <1>     add     byte [dma_mode], 10h ; = 58h
                            <1> sysdma_16:
00012A60 80E307             <1>     and     bl, 07h
00012A63 881D[F68C0100]     <1>     mov     [dma_channel], bl
00012A69 8915[008D0100]     <1>     mov     [dma_start], edx
00012A6F 890D[048D0100]     <1>     mov     [dma_count], ecx
                            <1>
                            <1>     ; 28/08/2017
                            <1>     ;call   dma_init
                            <1>     ;jmp    sysret
00012A75 E945010000         <1>     jmp     dma_init
                            <1>
                            <1> sysdma_5:
00012A7A 80FF05             <1>     cmp     bh, 5
00012A7D 726D               <1>     jb      short sysdma_3
00012A7F 7759               <1>     ja      short sysdma_6
                            <1>
                            <1>     ; Get the system's default dma buffer addr and size
00012A81 B800000500         <1>     mov     eax, sb16_dma_buffer
00012A86 B900000100         <1>     mov     ecx, 65536 ; Buffer size in bytes
                            <1>
                            <1> sysdma_51:
                            <1>     ; 0 = there is not a dma buffer (in use or available)
00012A8B A3[348E0100]       <1>     mov     [u.r0], eax
                            <1>
00012A90 8B2D[308E0100]     <1>     mov     ebp, [u.usp]  ; ebp points to user's registers
00012A96 894D18             <1>     mov     [ebp+24], ecx ; return to user with ecx value
                            <1>
00012A99 E9E09FFFFF         <1>     jmp     sysret
                            <1>
                            <1> sysdma_2:
                            <1>     ; Get current dma buffer offset (& page)
                            <1>     ; 28/08/2017
00012A9E 0FB635[F68C0100]   <1>     movzx   esi, byte [dma_channel]
00012AA5 0FB696[B8350100]   <1>     movzx   edx, byte [dma_flip+esi]
00012AAC EE                 <1>     out     dx, al                  ; flip-flop clear
00012AAD 8A96[90350100]     <1>     mov     dl, [dma_adr+esi]
00012AB3 EC                 <1>     in      al, dx                  ; get dma position
00012AB4 0FB6D8             <1>     movzx   ebx, al
```

```
4387 00012AB7 EC                  <1>     in      al, dx
4388 00012AB8 88C7                <1>     mov     bh, al
4389                              <1>
4390 00012ABA 6683FE04            <1>     cmp     si, 4   ; channel number ?
4391 00012ABE 7202                <1>     jb      short sysdma_21 ; 8 bit dma channel
4392                              <1>
4393 00012AC0 D1E3                <1>     shl     ebx, 1  ; word offset to byte offset
4394                              <1>
4395                              <1> sysdma_21:
4396 00012AC2 891D[348E0100]      <1>     mov     [u.r0], ebx
4397                              <1>
4398 00012AC8 8A96[A0350100]      <1>     mov     dl, [dma_page+esi]
4399 00012ACE EC                  <1>     in      al, dx                  ; get dma page
4400                              <1>
4401                              <1>     ;add    [u.ro+2], al
4402 00012ACF 0805[368E0100]      <1>     or      [u.r0+2], al
4403                              <1>
4404 00012AD5 E9A49FFFFF          <1>     jmp     sysret
4405                              <1>
4406                              <1> sysdma_6:
4407 00012ADA 80FF06              <1>     cmp     bh, 6
4408 00012ADD 775C                <1>     ja      short sysdma_7
4409                              <1>
4410                              <1>     ; 28/08/2017
4411                              <1>     ; Get current DMA buffer addr and size
4412 00012ADF A1[F88C0100]        <1>     mov     eax, [dma_addr] ; dma buffer address
4413 00012AE4 8B0D[FC8C0100]      <1>     mov     ecx, [dma_size] ; dma buffer size (in bytes)
4414                              <1>
4415 00012AEA EB9F                <1>     jmp     short sysdma_51
4416                              <1>
4417                              <1> sysdma_3:
4418 00012AEC 80FF03              <1>     cmp     bh, 3
4419 00012AEF 72AD                <1>     jb      short sysdma_2
4420 00012AF1 772F                <1>     ja      short sysdma_4
4421                              <1>
4422                              <1>     ; Get current dma count down value (remain bytes)
4423                              <1>     ; 28/08/2017
4424 00012AF3 0FB635[F68C0100]    <1>     movzx   esi, byte [dma_channel]
4425 00012AFA 0FB696[B8350100]    <1>     movzx   edx, byte [dma_flip+esi]
4426 00012B01 EE                  <1>     out     dx, al                  ; flip-flop clear
4427 00012B02 8A96[98350100]      <1>     mov     dl, [dma_cnt+esi] ; dma count register addr
4428 00012B08 EC                  <1>     in      al, dx
4429 00012B09 0FB6D8              <1>     movzx   ebx, al
4430 00012B0C EC                  <1>     in      al, dx
4431 00012B0D 88C7                <1>     mov     bh, al
4432                              <1>
4433 00012B0F 6683FE04            <1>     cmp     si, 4   ; channel number ?
4434 00012B13 7202                <1>     jb      short sysdma_31 ; 8 bit dma channel
4435                              <1>
4436 00012B15 D1E3                <1>     shl     ebx, 1  ; word count to byte count
4437                              <1>
4438                              <1> sysdma_31:
4439 00012B17 891D[348E0100]      <1>     mov     [u.r0], ebx
4440                              <1>
4441 00012B1D E95C9FFFFF          <1>     jmp     sysret
4442                              <1>
4443                              <1> sysdma_4:
4444                              <1>     ; Get current DMA channel number
4445                              <1>     ; 28/08/2017
4446 00012B22 8A25[F58C0100]      <1>     mov     ah, [dma_user]
4447 00012B28 20E4                <1>     and     ah, ah
4448 00012B2A 7539                <1>     jnz     short sysdma_42
4449                              <1>
4450                              <1> sysdma_41:
4451                              <1>     ; Not a valid dma channel (in use)
4452 00012B2C C705[348E0100]FFFF- <1>     mov     dword [u.r0], -1 ; 0FFFFFFFFh
4452 00012B34 FFFF                <1>
4453 00012B36 E9439FFFFF          <1>     jmp     sysret
4454                              <1>
4455                              <1> sysdma_7:
4456                              <1>     ; DMA service STOP
4457 00012B3B A0[858E0100]        <1>     mov     al, [u.uno]
4458 00012B40 3A05[F58C0100]      <1>     cmp     al, [dma_user]
4459 00012B46 7543                <1>     jne     short sysdma_72
4460                              <1>
4461 00012B48 28C0                <1>     sub     al, al ; 0
4462                              <1>
4463 00012B4A A2[F58C0100]        <1>     mov     [dma_user], al ; clear user
4464                              <1>
4465 00012B4F 8605[F78C0100]      <1>     xchg    al, [dma_mode]
4466 00012B55 20C0                <1>     and     al, al
4467                              <1>     ;jz     short sysdma_err
4468 00012B57 754E                <1>     jnz     short sysdma_73
4469                              <1>
4470                              <1> sysdma_71:
4471 00012B59 31C0                <1>     xor     eax, eax
4472 00012B5B A3[348E0100]        <1>     mov     [u.r0], eax; 0
4473 00012B60 E9199FFFFF          <1>     jmp     sysret
4474                              <1>
4475                              <1> sysdma_42:
4476 00012B65 8B35[F88C0100]      <1>     mov     esi, [dma_addr]
4477 00012B6B 21F6                <1>     and     esi, esi
4478 00012B6D 74BD                <1>     jz      short sysdma_41
4479                              <1>
4480 00012B6F 46                  <1>     inc     esi ; -1 -> 0
4481 00012B70 74BA                <1>     jz      short sysdma_41
4482                              <1>
4483 00012B72 A0[F68C0100]        <1>     mov     al, [dma_channel]
4484                              <1>
4485 00012B77 3A25[858E0100]      <1>     cmp     ah, [u.uno]
4486 00012B7D 7502                <1>     jne     short sysdma_43
4487                              <1>
4488 00012B7F 30E4                <1>     xor     ah, ah ; DMA channel in use by current user
4489                              <1>
4490                              <1> sysdma_43:
4491 00012B81 A3[348E0100]        <1>     mov     [u.r0], eax ; AL = dma channel number
4492                              <1>                         ; AH > 0 if the the channel
4493                              <1>                         ; in use by another user/process
4494 00012B86 E9F39EFFFF          <1>     jmp     sysret
4495                              <1>
4496                              <1> sysdma_72:
4497                              <1>     ; 28/08/2017
4498 00012B8B 803D[F58C0100]00    <1>     cmp     byte [dma_user], 0
4499 00012B92 76C5                <1>     jna     short sysdma_71 ; Nothing to do !
4500                              <1>
4501 00012B94 833D[F88C0100]00    <1>     cmp     dword [dma_addr], 0
4502                              <1>     ;ja     sysdma_0_err
4503                              <1>     ; 30/07/2022
4504 00012B9B 7605                <1>     jna     short sysdma_74
4505 00012B9D E9BBFDFFFF          <1>     jmp     sysdma_0_err
4506                              <1>
4507                              <1> sysdma_74:
4508 00012BA2 A2[F58C0100]        <1>     mov     [dma_user], al ; reset to current user
4509                              <1>
```

```
                            <1> sysdma_73:
                            <1>     ; 28/08/2017
00012BA7 0FB635[F68C0100]   <1>     movzx   esi, byte [dma_channel]
00012BAE 0FB696[A8350100]   <1>     movzx   edx, byte [dma_mask+esi]
00012BB5 A0[F68C0100]       <1>     mov     al, [dma_channel]
00012BBA 0C04               <1>     or      al, 4
00012BBC EE                 <1>     out     dx, al
                            <1>
00012BBD EB9A               <1>     jmp     short sysdma_71
                            <1>
                            <1> dma_init:
                            <1>     ; 30/07/2022 (TRDOS 386 Kernel v2.0.5)
                            <1>     ; 28/08/2017
                            <1>     ; 20/08/2017
                            <1>     ; DMA initialization
                            <1>     ; 14/08/2017
                            <1>     ; 03/08/2017, 06/08/2017, 08/08/2017
                            <1>     ; 02/07/2017, 13/07/2017, 16/07/2017, 30/07/2017
                            <1>     ; (Derived from 'DMA_INIT' procedure in SB16MOD.ASM)
                            <1>     ; Modified for TRDOS 386 DMA buffer allocation & initialization !
                            <1>
00012BBF 8B1D[008D0100]     <1>     mov     ebx, [dma_start]
00012BC5 8B0D[048D0100]     <1>     mov     ecx, [dma_count]
                            <1>
00012BCB 0FB635[F68C0100]   <1>     movzx   esi, byte [dma_channel]
                            <1>
00012BD2 6683FE04           <1>     cmp     si, 4
00012BD6 7204               <1>     jb      short gdmi1
                            <1>     ; 08/08/2017
                            <1>     ;shr    cx, 1 ; word count
                            <1>     ; 30/07/2022
00012BD8 D1E9               <1>     shr     ecx, 1
00012BDA D1EB               <1>     shr     ebx, 1 ; convert byte offset to word offset
                            <1> gdmi1:
                            <1>     ;mov    [dma_poff], bx ; 08/08/2017
                            <1>     ;dec    cx                      ; dma size = block size - 1
                            <1>     ; 30/07/2022
00012BDC 49                 <1>     dec     ecx
                            <1>
00012BDD 0FB696[A8350100]   <1>     movzx   edx, byte [dma_mask+esi] ; 30/07/2017
00012BE4 A0[F68C0100]       <1>     mov     al, [dma_channel]
00012BE9 0C04               <1>     or      al, 4
00012BEB EE                 <1>     out     dx, al                  ; dma channel mask
                            <1>
00012BEC 30C0               <1>     xor     al, al ; 0 ; any value ! 08/08/2017
00012BEE 8A96[B8350100]     <1>     mov     dl, [dma_flip+esi]
00012BF4 EE                 <1>     out     dx, al                  ; flip-flop clear
                            <1>
00012BF5 8A96[B0350100]     <1>     mov     dl, [dma_mod+esi]
00012BFB A0[F68C0100]       <1>     mov     al, [dma_channel]  ; 13/07/2017
00012C00 2403               <1>     and     al, 3
                            <1>     ; 08/08/2017
00012C02 0A05[F78C0100]     <1>     or      al, [dma_mode] ; 58h    ; dma mode for SB16
00012C08 EE                 <1>     out     dx, al
                            <1>
00012C09 8A96[90350100]     <1>     mov     dl, [dma_adr+esi]
00012C0F 88D8               <1>     mov     al, bl
00012C11 EE                 <1>     out     dx, al                  ; offset low
                            <1>
00012C12 88F8               <1>     mov     al, bh
00012C14 EE                 <1>     out     dx, al                  ; offset high
                            <1>
00012C15 8A96[98350100]     <1>     mov     dl, [dma_cnt+esi]
00012C1B 88C8               <1>     mov     al, cl
00012C1D EE                 <1>     out     dx, al                  ; size low
                            <1>
00012C1E 88E8               <1>     mov     al, ch
00012C20 EE                 <1>     out     dx, al                  ; size high
                            <1>
00012C21 8A96[A0350100]     <1>     mov     dl, [dma_page+esi]
                            <1>     ; 14/08/2017
00012C27 6683FE04           <1>     cmp     si, 4
00012C2B 7305               <1>     jnb     short gdmi2
00012C2D C1EB10             <1>     shr     ebx, 16
00012C30 EB06               <1>     jmp     short gdmi3
                            <1> gdmi2:
                            <1>     ; 09/08/2017
00012C32 C1EB0F             <1>     shr     ebx, 15  ; complete 16 bit shift
00012C35 80E3FE             <1>     and     bl, 0FEh ; clear bit 0 (not necessary)
                            <1> gdmi3:
00012C38 88D8               <1>     mov     al, bl
00012C3A EE                 <1>     out     dx, al                  ; page
                            <1>
00012C3B 8A96[A8350100]     <1>     mov     dl, [dma_mask+esi]
00012C41 A0[F68C0100]       <1>     mov     al, [dma_channel]  ; 13/07/2017
00012C46 2403               <1>     and     al, 3
00012C48 EE                 <1>     out     dx, al                  ; dma channel unmask
                            <1>
                            <1>     ;retn
                            <1>     ; 28/08/2017
00012C49 E9309EFFFF         <1>     jmp     sysret
                            <1>
                            <1> sysstdio: ; STDIN/STDOUT/STDERR functions
                            <1>     ; 18/09/2024
                            <1>     ; 07/09/2024
                            <1>     ;       (STDAUX/STDPRN functions, pre-definions)
                            <1>     ;       (!these functions are not ready!)
                            <1>     ; 24/08/2024
                            <1>     ; 23/08/2024
                            <1>     ; 20/08/2024 - TRDOS 386 Kernel v2.0.9
                            <1>     ;
                            <1>     ; Inputs:
                            <1>     ;       BL = 0 -> read a character on stdin (wait)
                            <1>     ;       BL = 1 -> read a character on stdin (no wait)
                            <1>     ;       BL = 2 -> write a character onto stdout (redirection)
                            <1>     ;       BL = 3 -> write a character onto stderr (no redirection)
                            <1>     ;       BL = 4 -> redirect stdin to file (if cl > 0)
                            <1>     ;       BL = 5 -> redirect stdout to file (if cl > 0)
                            <1>     ;       BL = 6 -> read character (ascii and scancode) on stdin
                            <1>     ;                                 -no redirection, wait-
                            <1>     ;       BL = 7 -> read character (ascii and scancode) on stdin
                            <1>     ;                                 -no redirection, no wait-
                            <1>     ;       BL = 8 -> write character and color onto stdout
                            <1>     ;                                 -no redirection-
                            <1>     ;       BL = 9 -> ungetchar (put back the ascii code in u.getc)
                            <1>     ;
                            <1>     ;       For BL=2,3,8,9
                            <1>     ;           CL = character (ascii) code
                            <1>     ;       For BL=8
                            <1>     ;           CH = color (Attribute) -CGA-
                            <1>     ;       For BL=4,5
                            <1>     ;           CL = file descriptor number + 1
                            <1>     ;           (File must be open and that number is 'u.fp' index)
                            <1>     ;
```

```
4634                                <1>     ;       07/09/2024 (these subfunctions will not be handled
4635                                <1>     ;                 by kernel v2.0.9 for now)
4636                                <1>     ;        -I am writing them here for C compiler compatibility,
4637                                <1>     ;         for now. Ref: SCC STDIO.H-
4638                                <1>     ;       BL = 10 -> get STDAUX status
4639                                <1>     ;       BL = 11 -> get/select STDAUX (COM) port
4640                                <1>     ;                            (clears redirection)
4641                                <1>     ;       BL = 12 -> redirect STDAUX
4642                                <1>     ;       BL = 13 -> STDAUX IOCTL (control functions)
4643                                <1>     ;       BL = 14 -> read byte/character from STDAUX
4644                                <1>     ;       BL = 15 -> write byte/character to STDAUX
4645                                <1>     ;       BL = 16 -> get STDPRN status (LPT printer only)
4646                                <1>     ;       BL = 17 -> redirect STDPRN
4647                                <1>     ;       BL = 18 -> STDPRN IOCTL (init/control functions)
4648                                <1>     ;       BL = 19 -> write byte/character to STDPRN
4649                                <1>     ;
4650                                <1>     ;       For BL = 11
4651                                <1>     ;           If CL = 0 -> get STDAUX (COM) port number
4652                                <1>     ;              CL = 1 -> COM1
4653                                <1>     ;              CL = 2 -> COM2
4654                                <1>     ;              CL > 4 -> invalid
4655                                <1>     ;       For BL = 12
4656                                <1>     ;           If ECX = 0 -> clear redirection
4657                                <1>     ;           If  CH & 7Fh = 0 -> redirect INPUT
4658                                <1>     ;               CH & 7Fh > 0 -> redirect OUTPUT
4659                                <1>     ;           If  CH & 80h = 80h
4660                                <1>     ;              CL = file handle (descriptor index)
4661                                <1>     ;           If  CH & 80h = 0
4662                                <1>     ;              CL = console (pseudo) tty number,
4663                                <1>     ;                   1 to 8 for tty0 to tty7.
4664                                <1>     ;       For BL = 13
4665                                <1>     ;           If CL = 0 -> get control byte
4666                                <1>     ;           If CL > 1 -> set control byte (in CL)
4667                                <1>     ;              bit 0 = 1
4668                                <1>     ;              bit 1 = data bits (set=8, clear=7)
4669                                <1>     ;              bit 2 = stop bit (set=2, clear=1)
4670                                <1>     ;              bit 3 = parity bit (set=yes, clear=no)
4671                                <1>     ;              bit 4 = parity bit (set=even, clear=odd)
4672                                <1>     ;              bit 5-6-7 = data rate
4673                                <1>     ;                      000 = 9600
4674                                <1>     ;                       001 = 19200
4675                                <1>     ;                      010 = 38400
4676                                <1>     ;                      011 = 57600
4677                                <1>     ;                      100 = 115200
4678                                <1>     ;       For BL = 15, 19
4679                                <1>     ;           CL = byte/character to r/w
4680                                <1>     ;       For BL = 17
4681                                <1>     ;           If ECX = 0 -> clear redirection
4682                                <1>     ;           If  CH & 80h = 80h
4683                                <1>     ;              CL = file handle (descriptor index)
4684                                <1>     ;           If  CH & 80h = 0
4685                                <1>     ;              CL = console (pseudo) tty number,
4686                                <1>     ;                   1 to 8 for tty0 to tty7.
4687                                <1>     ;       For BL = 18
4688                                <1>     ;           If CL = 0 -> initialize printer (command=0)
4689                                <1>     ;              CL > 0 -> configuration/command byte
4690                                <1>     ;                        (reserved, not used)
4691                                <1>     ;
4692                                <1>     ; Outputs:
4693                                <1>     ;
4694                                <1>     ;       For BL=0,1,2,3,6,7,8,9
4695                                <1>     ;           AL = character (ascii) code
4696                                <1>     ;       For BL=6,7
4697                                <1>     ;           AH = scan code
4698                                <1>     ;       For BL=4,5
4699                                <1>     ;           AL = file descriptor (CL input - 1)
4700                                <1>     ;
4701                                <1>     ;       If CF=1
4702                                <1>     ;          AL (EAX) = error code
4703                                <1>     ;          If EAX = 0 -> EOF ; 18/09/2024
4704                                <1>     ;
4705                                <1>     ;       07/09/2024
4706                                <1>     ;       For BL = 10 to 19
4707                                <1>     ;           IF CF = 1, EAX/AL = error code
4708                                <1>     ;     .. If CF = 0 ...
4709                                <1>     ;       For BL = 10 & 16
4710                                <1>     ;           AL (EAX) = status
4711                                <1>     ;       For BL = 11
4712                                <1>     ;           EAX/AL = COM (serial) port (1 to 4)
4713                                <1>     ;       For BL = 12, 15, 17, 18, 19
4714                                <1>     ;           EAX/AL = 0
4715                                <1>     ;       For BL = 13
4716                                <1>     ;           If CL input = 0
4717                                <1>     ;                EAX/AL = current ctrl/cfg byte
4718                                <1>     ;            If CL input > 0
4719                                <1>     ;                EAX/AL = ctrl/cfg byte
4720                                <1>     ;       For BL = 14
4721                                <1>     ;           EAX/AL = byte/character
4722                                <1>     ;
4723                                <1>
4724 00012C4E 80FB0A              <1>     cmp     bl, (end_of_stdiofuncs-stdiofuncs)>>2
4725 00012C51 7214                <1>     jb      short sysstdio_0
4726                                <1>
4727                                <1>     ; invalid parameter (invalid sub function number)
4728 00012C53 B817000000          <1>     mov     eax, ERR_INV_PARAMETER
4729                                <1> sysstdio_err:
4730 00012C58 A3[348E0100]        <1>     mov     [u.r0], eax
4731 00012C5D A3[A08E0100]        <1>     mov     [u.error], eax
4732 00012C62 E9F79DFFFF          <1>     jmp     error
4733                                <1>
4734                                <1> sysstdio_0:
4735 00012C67 31C0                <1>     xor     eax, eax ; 0
4736 00012C69 A3[A08E0100]        <1>     mov     [u.error], eax ; clear previous error code
4737 00012C6E A3[348E0100]        <1>     mov     [u.r0], eax ; clear previous return code
4738                                <1>
4739 00012C73 0FB6F3              <1>     movzx   esi, bl
4740 00012C76 C1E602              <1>     shl     esi, 2
4741 00012C79 FFA6[7F2C0100]      <1>     jmp     dword [esi+stdiofuncs]
4742                                <1>
4743                                <1> stdiofuncs:
4744 00012C7F [092D0100]          <1>     dd      readstdinw
4745 00012C83 [092D0100]          <1>     dd      readstdinnw
4746 00012C87 [202E0100]          <1>     dd      writestdout
4747 00012C8B [D72D0100]          <1>     dd      writestderr
4748 00012C8F [A72C0100]          <1>     dd      redirstdin
4749 00012C93 [E52C0100]          <1>     dd      redirstdout
4750                                <1> stdinacsc:
4751 00012C97 [F52D0100]          <1>     dd      readstdin2w
4752 00012C9B [F52D0100]          <1>     dd      readstdin2nw
4753                                <1> stdoutcc:
4754 00012C9F [2B2E0100]          <1>     dd      writestdoutcc
4755 00012CA3 [422E0100]          <1>     dd      ungetchar
4756                                <1> end_of_stdiofuncs:
4757                                <1>
```

```
                                <1> redirstdin:
                                <1>     ; CL = file handle (descriptor/index) + 1
00012CA7 08C9                   <1>     or      cl, cl
00012CA9 7508                   <1>     jnz     short redirstdin_0
00012CAB 880D[9C8E0100]         <1>     mov     [u.stdin], cl ; 0 ; reset STDIN to keyboard
00012CB1 EB24                   <1>     jmp     short redirstdin_3
                                <1> redirstdin_0:
00012CB3 88C8                   <1>     mov     al, cl
00012CB5 E8D5D5FFFF             <1>     call    getf
                                <1>     ; eax = first cluster
                                <1>     ; ebx = system (not user) open file index
00012CBA 80BB[E4830100]01       <1>     cmp     byte [ebx+OF_MODE], 1 ; open for read
00012CC1 7407                   <1>     je      short redirstdin_1
                                <1> redir_acc_denied:
00012CC3 B805000000             <1>     mov     eax, ERR_ACCESS_DENIED
00012CC8 EB8E                   <1>     jmp     short sysstdio_err
                                <1> redirstdin_1:
00012CCA 880D[9C8E0100]         <1>     mov     [u.stdin], cl
                                <1> redirstdout_2:
00012CD0 49                     <1>     dec     ecx
00012CD1 880D[348E0100]         <1>     mov     [u.r0], cl ; return file descriptor
                                <1> redirstdin_3:
                                <1> redirstdout_3:
00012CD7 66C705[9E8E0100]00-    <1>     mov     word [u.ungetc], 0
00012CDF 00                     <1>
                                <1>             ; reset u.ungetc and u.getc
00012CE0 E9999DFFFF             <1>     jmp     sysret
                                <1>
                                <1> redirstdout:
                                <1>     ; CL = file handle (descriptor/index) + 1
00012CE5 08C9                   <1>     or      cl, cl
00012CE7 7508                   <1>     jnz     short redirstdout_0
00012CE9 880D[9D8E0100]         <1>     mov     [u.stdout], cl ; 0 ; reset STDOUT to keyboard
00012CEF EBE6                   <1>     jmp     short redirstdout_3
                                <1> redirstdout_0:
00012CF1 88C8                   <1>     mov     al, cl
00012CF3 E897D5FFFF             <1>     call    getf
                                <1>     ; eax = first cluster
                                <1>     ; ebx = system (not user) open file index
00012CF8 80BB[E4830100]02       <1>     cmp     byte [ebx+OF_MODE], 2 ; open for write
00012CFF 75C2                   <1>     jne     short redir_acc_denied
                                <1> redirstdout_1:
00012D01 880D[9D8E0100]         <1>     mov     [u.stdout], cl
00012D07 EBC7                   <1>     jmp     short redirstdout_2
                                <1>
                                <1> readstdinw:
                                <1> readstdinnw:
                                <1>     ; read a character on stdin (wait)
                                <1>     ;sub    eax, eax
00012D09 3805[9E8E0100]         <1>     cmp     byte [u.ungetc], al ; 0
00012D0F 761E                   <1>     jna     short readstdinw_0
                                <1>     ; read the character in u.getc buffer (put by ungetchar)
00012D11 A2[9E8E0100]           <1>     mov     [u.ungetc], al ; 0
00012D16 8605[9F8E0100]         <1>     xchg    al, [u.getc]
                                <1> readstdinw_retn:
00012D1C A2[348E0100]           <1>     mov     [u.r0], al
00012D21 E9589DFFFF             <1>     jmp     sysret
                                <1>
                                <1> readstdinw_3:
                                <1>     ; 23/08/2024
00012D26 B410                   <1>     mov     ah, 10h
00012D28 E8EFE1FEFF             <1>     call    int16h
00012D2D EBED                   <1>     jmp     short readstdinw_retn
                                <1>
                                <1> readstdinw_0:
00012D2F A0[9C8E0100]           <1>     mov     al, [u.stdin]
00012D34 08C0                   <1>     or      al, al
00012D36 745F                   <1>     jz      short readstdinw_1
                                <1>
                                <1> readstdinf:
                                <1>     ; 24/08/2024
00012D38 48                     <1>     dec     eax
                                <1>
                                <1>     ; file
00012D39 89C3                   <1>     mov     ebx, eax ; File handle (descriptor/index)
                                <1>     ; 18/09/2024
00012D3B B9[5A2E0100]           <1>     mov     ecx, charbuf ; buffer address
00012D40 BA01000000             <1>     mov     edx, 1  ; 1 character only
                                <1>
                                <1>     ; 18/09/2024
                                <1>     ;;;
00012D45 E847D5FFFF             <1>     call    getf1
00012D4A 722B                   <1>     jc      short redirect_fno_err
                                <1>
                                <1> readstdinf_@:
                                <1>     ; eax = first cluster
                                <1>     ; ebx = file descriptor (system, open files)
                                <1>
00012D4C E8A4A4FFFF             <1>     call    rw1
00012D51 7229                   <1>     jc      short redirection_err
                                <1>
00012D53 FE05[9A8E0100]         <1>     inc     byte [u.kcall]
                                <1>     ;mov    byte [u.kcall], 1 ; >0 means system buffer
                                <1>                             ; (buff addr is not virtual)
00012D59 E8F7D8FFFF             <1>     call    readi
                                <1>
                                <1> redirected_rw_OK:
00012D5E 31C0                   <1>     xor     eax, eax
00012D60 3905[608E0100]         <1>     cmp     [u.nread], eax ; 0
00012D66 7623                   <1>     jna     short readstdinf_EOF
                                <1>
00012D68 A0[5A2E0100]           <1>     mov     al, [charbuf]
                                <1> ;redirected_w_not_OK:
00012D6D A3[348E0100]           <1>     mov     [u.r0], eax
00012D72 E9079DFFFF             <1>     jmp     sysret
                                <1>
                                <1> redirect_fno_err:
00012D77 B80A000000             <1>     mov     eax, ERR_FILE_NOT_OPEN ; file not open !
                                <1> redirection_err:
00012D7C A3[A08E0100]           <1>     mov     [u.error], eax
                                <1> readstdinf_EOF_@:
00012D81 A3[348E0100]           <1>     mov     [u.r0], eax ; error code ; 0 = EOF
00012D86 E9D39CFFFF             <1>     jmp     error
                                <1>
                                <1> readstdinf_EOF:
                                <1>     ; 'end of file !' error
00012D8B C705[A08E0100]1000-    <1>     mov     dword [u.error], ERR_FILE_EOF ; 16
00012D93 0000                   <1>
00012D95 EBEA                   <1>     jmp     short readstdinf_EOF_@
                                <1>     ;;;
                                <1>
                                <1> readstdinw_1:
                                <1>     ; 18/09/2024
00012D97 20DB                   <1>     and     bl, bl
```

```
4880 00012D99 748B              <1>     jz      short readstdinw_3 ; wait (int16h, 10h)
4881                            <1>     ; no wait (int16h, 11h)
4882 00012D9B B411              <1>     mov     ah,11h  ; EXTENDED ASCII STATUS
4883 00012D9D E87AE1FEFF        <1>     call    int16h
4884                            <1>     ; ah = scan code, al = ascii code
4885                            <1>     ;jnz    short readstdinw_retn
4886                            <1>     ; 23/08/2024
4887 00012DA2 7582              <1>     jnz     short readstdinw_3
4888                            <1> readstdinw_2:
4889                            <1>
4890                            <1> ; 23/08/2024
4891                            <1> ;   ; if zf=1 at here
4892                            <1> ;   ; it means 'no code available' for function 11h
4893                            <1> ;   and     bl, bl  ; 0 ?
4894                            <1> ;   jz      short  short readstdinw_retn ; function 10h
4895                            <1> ;   ; function 11h
4896                            <1> ;   ; [u.r0] = 0 = eax return
4897                            <1> readstdin2w@_retn:
4898 00012DA4 E9D59CFFFF        <1>     jmp     sysret
4899                            <1>
4900                            <1> writestdoutf:
4901                            <1>     ; 24/08/2024
4902 00012DA9 48                <1>     dec     eax
4903                            <1>
4904                            <1>     ; file
4905 00012DAA 89C3              <1>     mov     ebx, eax ; File handle (descriptor/index)
4906                            <1>     ; 18/09/2024
4907 00012DAC 880D[5A2E0100]    <1>     mov     byte [charbuf], cl
4908 00012DB2 B9[5A2E0100]      <1>     mov     ecx, charbuf
4909 00012DB7 BA01000000        <1>     mov     edx, 1  ; 1 character only
4910                            <1>
4911                            <1>     ; 18/09/2024
4912                            <1>     ;;;
4913 00012DBC E8D0D4FFFF        <1>     call    getf1
4914 00012DC1 72B4              <1>     jc      short redirect_fno_err
4915                            <1>
4916                            <1> writestdoutf_@:
4917                            <1>     ; eax = first cluster
4918                            <1>     ; ebx = file descriptor (system, open files)
4919                            <1>
4920 00012DC3 E82DA4FFFF        <1>     call    rw1
4921 00012DC8 72B2              <1>     jc      short redirection_err
4922                            <1>
4923 00012DCA FE05[9A8E0100]    <1>     inc     byte [u.kcall]
4924                            <1>     ;mov    byte [u.kcall], 1 ; >0 means system buffer
4925                            <1>                               ; (buff addr is not virtual)
4926 00012DD0 E859DFFFFF        <1>     call    writei
4927                            <1>
4928                            <1>     ;xor    eax, eax
4929                            <1>     ;cmp    [u.nread], eax ; 0
4930                            <1>     ;jna    short redirected_w_not_OK
4931 00012DD5 EB87              <1>     jmp     short redirected_rw_OK
4932                            <1>     ;;;
4933                            <1>
4934                            <1> writestderr:        ; skip redirection
4935                            <1>             ; STDERR = STDOUT (as file redirection disabled)
4936 00012DD7 88C8              <1>     mov     al, cl  ; character
4937 00012DD9 A2[348E0100]      <1>     mov     [u.r0], al
4938 00012DDE B40E              <1>     mov     ah, 0Eh ; write a character (as tty write)
4939 00012DE0 BB07000000        <1>     mov     ebx, 07h ; video page 0 (and color/attrib 07h)
4940                            <1>     ; 23/08/2024
4941 00012DE5 8A3D[AE760100]    <1>     mov     bh, [ptty] ; ACTIVE_PAGE
4942 00012DEB E834E9FEFF        <1>     call    _int10h
4943 00012DF0 E9899CFFFF        <1>     jmp     sysret
4944                            <1>
4945                            <1> readstdin2w:
4946                            <1> readstdin2nw:
4947 00012DF5 66C705[9E8E0100]00- <1>     mov     word [u.ungetc], 0
4947 00012DFD 00                <1>
4948                            <1>             ; reset u.ungetc and u.getc
4949 00012DFE B410              <1>     mov     ah, 10h  ; Keyboard, EXTENDED READ
4950                            <1>     ;cmp    bl, 6
4951 00012E00 80FB06            <1>     cmp     bl, (stdinacsc-stdiofuncs)>>2
4952 00012E03 740B              <1>     je      short readstdin2w_1 ; wait (int16h, 10h)
4953                            <1>     ; no wait (int16h, 11h)
4954 00012E05 FEC4              <1>     inc     ah  ; function 11h ; EXTENDED ASCII STATUS
4955                            <1>
4956                            <1>     ; 24/08/2024
4957 00012E07 E810E1FEFF        <1>     call    int16h
4958                            <1>     ; ah = scan code, al = ascii code
4959                            <1>     ;jnz    short readstdin2w_retn
4960                            <1>     ; 23/08/2024
4961 00012E0C 7496              <1>     jz      short readstdin2w@_retn
4962                            <1> ;readstdin2w_1:
4963                            <1>
4964                            <1> ; 23/08/2024
4965                            <1> ;   ; if zf=1 at here
4966                            <1> ;   ; it means 'no code available' for function 11h
4967                            <1> ;   ;cmp    bl, 6
4968                            <1> ;   cmp     bl, (stdinacsc-stdiofuncs)>>2
4969                            <1> ;   ;je     short readstdin2w_retn ; function 10h
4970                            <1> ;   ; function 11h
4971                            <1> ;   ; [u.r0] = 0 = eax return
4972                            <1> ;   ;jmp    sysret
4973                            <1> ;   jne     short readstdin2w@_retn
4974                            <1>
4975                            <1>     ; 23/08/2024
4976 00012E0E B410              <1>     mov     ah, 10h
4977                            <1> readstdin2w_1:      ; 18/09/2024
4978 00012E10 E807E1FEFF        <1>     call    int16h
4979                            <1>
4980                            <1> readstdin2w_retn:
4981 00012E15 66A3[348E0100]    <1>     mov     [u.r0], ax
4982 00012E1B E95E9CFFFF        <1>     jmp     sysret
4983                            <1>
4984                            <1> writestdout:
4985                            <1>     ; 18/09/2024
4986 00012E20 A0[9D8E0100]      <1>     mov     al, [u.stdout]
4987 00012E25 20C0              <1>     and     al, al
4988 00012E27 7580              <1>     jnz     short writestdoutf ; 18/09/2024
4989                            <1>
4990                            <1> writestdout_1:
4991                            <1>     ;mov    byte [ccolor], 07h ; default color (CGA)
4992                            <1>                                ; (black background, light gray character)
4993                            <1>     ; 18/09/2024
4994 00012E29 B507              <1>     mov     ch, 07h
4995                            <1> writestdoutcc:
4996 00012E2B BE[348E0100]      <1>     mov     esi, u.r0 ; buffer
4997                            <1>     ; 18/09/2024
4998 00012E30 880E              <1>     mov     [esi], cl ; character
4999                            <1>     ;;cmp   bl, 8 ; write char and color
5000                            <1>     ;cmp    bl, (stdoutcc-stdiofuncs)>>2
5001                            <1>     ;jne    short writestdout_2
5002 00012E32 882D[AF760100]    <1>     mov     byte [ccolor], ch ; color/attribute (CGA)
```

```
5003                                <1> writestdout_2:
5004 00012E38 E8CBD6FFFF            <1>     call    print_cmsg
5005                                <1>     ; 18/09/2024
5006                                <1>     ;mov    byte [ccolor], 07h ; set default color again
5007                                <1>
5008                                <1>     ; [u.r0] = written character (AL)
5009 00012E3D E93C9CFFFF            <1>     jmp     sysret
5010                                <1>
5011                                <1> ungetchar:
5012                                <1>     ; put a character back in u.getc STDIN buffer
5013 00012E42 C605[9E8E0100]01      <1>     mov     byte [u.ungetc], 1
5014 00012E49 880D[9F8E0100]        <1>     mov     [u.getc], cl
5015 00012E4F 880D[348E0100]        <1>     mov     [u.r0],cl
5016 00012E55 E9249CFFFF            <1>     jmp     sysret
5017                                <1>
5018                                <1>     ; 18/09/2024
5019 00012E5A 00                    <1> charbuf: db 0
5020                                <1>
5021                                <1> otty:
5022                                <1> sret:
5023                                <1> ocvt:
5024                                <1> ctty:
5025                                <1> cret:
5026                                <1> ccvt:
5027                                <1> rtty:
5028                                <1> wtty:
5029                                <1> rmem:
5030                                <1> wmem:
5031                                <1> rfd:
5032                                <1> rhd:
5033                                <1> wfd:
5034                                <1> whd:
5035                                <1> rlpt:
5036                                <1> wlpt:
5037                                <1> rcvt:
5038                                <1> xmtt:
5039 00012E5B C3                    <1>     retn
3440                                %include 'trdosk9.s' ; 04/01/2016
 1                                  <1> ; ****************************************************************************
 2                                  <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.10) - INITIALIZED DATA : trdosk9.s
 3                                  <1> ; ----------------------------------------------------------------------------
 4                                  <1> ; Last Update: 28/12/2025 (Previous: 29/12/2024 - Kernel v2.0.9)
 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 00012E5C 01                      <1>             db 1 ; A: = 0 & B: = 1
22                                  <1>
23                                  <1> Restore_CDIR:
24 00012E5D FF                      <1>             db 0FFh ; Initial value -> any number except 0
25                                  <1>
26                                  <1> msg_CRLF_temp:
27 00012E5E 070D0A00                <1>             db 07h, 0Dh, 0Ah, 0
28                                  <1>
29                                  <1> Magic_Bytes:
30 00012E62 04                      <1>             db 4
31 00012E63 01                      <1>             db 1
32                                  <1> mainprog_Version:
33 00012E64 07                      <1>             db 7
34 00012E65 5B5452444F535D204D-     <1>             db "[TRDOS] Main Program v2.0.10 (28/12/2025)"
34 00012E6E 61696E2050726F6772-     <1>
34 00012E77 616D2076322E302E31-     <1>
34 00012E80 30202832382F31322F-     <1>
34 00012E89 3230323529              <1>
35 00012E8E 0D0A                    <1>             db 0Dh, 0Ah
36 00012E90 286329204572646F67-     <1>             db "(c) Erdogan Tan 2005-2025"
36 00012E99 616E2054616E203230-     <1>
36 00012EA2 30352D32303235          <1>
37 00012EA9 0D0A00                  <1>             db 0Dh, 0Ah, 0
38                                  <1>
39                                  <1> MainProgCfgFile: ; 14/04/2016
40 00012EAC 4D41494E50524F472E-     <1>             db "MAINPROG.CFG", 0
40 00012EB5 43464700                <1>
41                                  <1>
42                                  <1> TRDOSPromptLabel:
43 00012EB9 5452444F53              <1>             db "TRDOS"
44 00012EBE 00                      <1>             db 0
45 00012EBF 00<rep 5h>              <1>                 times 5 db 0
46 00012EC4 00                      <1>             db 0
47                                  <1>
48                                  <1> ; INTERNAL COMMANDS
49                                  <1> Command_List:
50 00012EC5 44495200                <1> Cmd_Dir:    db "DIR", 0
51 00012EC9 434400                  <1> Cmd_Cd:             db "CD", 0
52 00012ECC 433A00                  <1> Cmd_Drive:  db "C:", 0
53 00012ECF 56455200                <1> Cmd_Ver:    db "VER", 0
54 00012ED3 4558495400              <1> Cmd_Exit:   db "EXIT", 0
55 00012ED8 50524F4D505400          <1> Cmd_Prompt: db "PROMPT", 0
56 00012EDF 564F4C554D4500          <1> Cmd_Volume: db "VOLUME", 0
57 00012EE6 4C4F4E474E414D4500      <1> Cmd_LongName:       db "LONGNAME", 0
58 00012EEF 4441544500              <1> Cmd_Date:   db "DATE", 0
59 00012EF4 54494D4500              <1> Cmd_Time:   db "TIME", 0
60 00012EF9 52554E00                <1> Cmd_Run:    db "RUN", 0
61 00012EFD 53455400                <1> Cmd_Set:    db "SET", 0
62 00012F01 434C5300                <1> Cmd_Cls:    db "CLS", 0
63 00012F05 53484F5700              <1> Cmd_Show:   db "SHOW", 0
64 00012F0A 44454C00                <1> Cmd_Del:    db "DEL", 0
65 00012F0E 41545452494200          <1> Cmd_Attrib: db "ATTRIB", 0
66 00012F15 52454E414D4500          <1> Cmd_Rename: db "RENAME", 0
67 00012F1C 524D44495200            <1> Cmd_Rmdir:  db "RMDIR", 0
68 00012F22 4D4B44495200            <1> Cmd_Mkdir:  db "MKDIR", 0
69 00012F28 434F505900              <1> Cmd_Copy:   db "COPY", 0
70 00012F2D 4D4F564500              <1> Cmd_Move:   db "MOVE", 0
71 00012F32 5041544800              <1> Cmd_Path:   db "PATH", 0
72 00012F37 4D454D00                <1> Cmd_Mem:    db "MEM", 0
73 00012F3B 00                      <1>             db 0
74 00012F3C 46494E4400              <1> Cmd_Find:   db "FIND", 0
75 00012F41 4543484F00              <1> Cmd_Echo:   db "ECHO", 0
76 00012F46 2A00                    <1> Cmd_Remark: db "*", 0
77 00012F48 3F00                    <1> Cmd_Help:   db "?", 0
78 00012F4A 44455649434500          <1> Cmd_Device: db "DEVICE", 0
79 00012F51 4445564C49535400        <1> Cmd_DevList:        db "DEVLIST", 0
```

```
 80 00012F59 434844495200        <1> Cmd_Chdir:  db "CHDIR", 0
 81 00012F5F 4245455000          <1> Cmd_Beep:   db "BEEP", 0
 82                              <1>
 83 00012F64 00                  <1>             db 0
 84                              <1>
 85                              <1> ; 15/02/2016 (FILE.ASM, 09/10/2011)
 86                              <1> invalid_fname_chars:
 87 00012F65 222728292A2B2C2F    <1>             db 22h, 27h, 28h, 29h, 2Ah, 2Bh, 2Ch, 2Fh
 88 00012F6D 3A3B3C3D3E3F40      <1>             db 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh, 40h
 89 00012F74 5B5C5D5E60          <1>             db 5Bh, 5Ch, 5Dh, 5Eh, 60h
 90                              <1> sizeInvFnChars  equ ($ - invalid_fname_chars)
 91                              <1> ;
 92                              <1>
 93                              <1> Msg_Enter_Date:
 94 00012F79 456E746572206E6577- <1>                 db 'Enter new date (dd-mm-yy): '
 94 00012F82 206461746520286464- <1>
 94 00012F8B 2D6D6D2D7979293A20  <1>
 95 00012F94 00                  <1>                 db 0
 96                              <1> Msg_Show_Date:
 97 00012F95 43757272656E742064- <1>                 db   'Current date is '
 97 00012F9E 61746520697320      <1>
 98 00012FA5 30                  <1> Day:            db   '0'
 99 00012FA6 30                  <1>             db   '0'
100 00012FA7 2F                  <1>                 db   '/'
101 00012FA8 30                  <1> Month:          db   '0'
102 00012FA9 30                  <1>             db   '0'
103 00012FAA 2F                  <1>                 db   '/'
104 00012FAB 30                  <1> Century:        db   '0'
105 00012FAC 30                  <1>                 db   '0'
106 00012FAD 30                  <1> Year:           db   '0'
107 00012FAE 30                  <1>             db   '0'
108 00012FAF 0D0A00              <1>                 db   0Dh, 0Ah, 0
109                              <1>
110                              <1> Msg_Enter_Time:
111 00012FB2 456E746572206E6577- <1>             db 'Enter new time: '
111 00012FBB 2074696D653A20      <1>
112 00012FC2 00                  <1>             db 0
113                              <1> Msg_Show_Time:
114 00012FC3 43757272656E742074- <1>             db   'Current time is '
114 00012FCC 696D6520697320      <1>
115 00012FD3 30                  <1> Hour:           db   '0'
116 00012FD4 30                  <1>             db   '0'
117 00012FD5 3A                  <1>             db   ':'
118 00012FD6 30                  <1> Minute:         db   '0'
119 00012FD7 30                  <1>             db   '0'
120 00012FD8 3A                  <1>             db   ':'
121 00012FD9 30                  <1> Second:         db   '0'
122 00012FDA 30                  <1>             db   '0'
123 00012FDB 0D0A00              <1>             db   0Dh, 0Ah, 0
124                              <1>
125                              <1> ;VolSize_Unit1:   dd 0
126                              <1> ;VolSize_Unit2:   dd 0
127                              <1>
128                              <1> VolSize_KiloBytes:
129 00012FDE 206B696C6F62797465- <1>             db " kilobytes", 0Dh, 0Ah, 0
129 00012FE7 730D0A00            <1>
130                              <1> VolSize_Bytes:
131 00012FEB 2062797465730D0A00  <1>             db " bytes", 0Dh, 0Ah, 0
132                              <1> Volume_in_drive:
133 00012FF4 0D0A                <1>             db 0Dh, 0Ah
134                              <1> Vol_FS_Name:
135 00012FF6 54522046533120      <1>             db "TR FS1 "
136 00012FFD 566F6C756D6520696E- <1>             db "Volume in drive "
136 00013006 20647269766520      <1>
137 0001300D 30                  <1> Vol_Drv_Name:   db 30h
138 0001300E 3A                  <1>             db ":"
139 0001300F 20697320            <1>             db " is "
140 00013013 0D0A00              <1>             db 0Dh, 0Ah, 0
141                              <1> Dir_Drive_Str:
142 00013016 54522D444F53204472- <1>                 db "TR-DOS Drive "
142 0001301F 69766520            <1>
143                              <1> Dir_Drive_Name:
144 00013023 303A                <1>                 db "0:"
145 00013025 0D0A                <1>                 db  0Dh, 0Ah
146                              <1> Vol_Str_Header:
147 00013027 566F6C756D65204E61- <1>                 db "Volume Name: "
147 00013030 6D653A20            <1>
148                              <1> Vol_Name:
149 00013034 00<rep 40h>         <1>             times 64 db 0
150 00013074 00                  <1>             db 0
151                              <1> Vol_Serial_Header:
152 00013075 0D0A                <1>             db 0Dh, 0Ah
153 00013077 566F6C756D65205365- <1>             db "Volume Serial No: "
153 00013080 7269616C204E6F3A20  <1>
154                              <1> Vol_Serial1:
155 00013089 30303030            <1>             db "0000"
156 0001308D 2D                  <1>             db "-"
157                              <1> Vol_Serial2:
158 0001308E 30303030            <1>             db "0000"
159 00013092 0D0A00              <1>             db 0Dh, 0Ah, 0
160                              <1>
161                              <1> ;Vol_Tot_Sec_Str_Start:
162                              <1> ;           dd 0
163                              <1> Vol_Total_Sector_Header:
164 00013095 0D0A                <1>             db 0Dh, 0Ah
165 00013097 566F6C756D65205369- <1>             db "Volume Size : ", 0
165 000130A0 7A65203A2000        <1>
166                              <1> ;Vol_Tot_Sec_Str:
167                              <1> ;           db "0000000000"
168                              <1> ;Vol_Tot_Sec_Str_End:
169                              <1> ;           db 0
170                              <1> ;Vol_Free_Sectors_Str_Start:
171                              <1> ;           dd 0
172                              <1> Vol_Free_Sectors_Header:
173 000130A6 467265652053706163- <1>             db "Free Space  : ", 0
173 000130AF 6520203A2000        <1>
174                              <1> ;Vol_Free_Sectors_Str:
175                              <1> ;           db "0000000000"
176                              <1> ;Vol_Free_Sectors_Str_End:
177                              <1> ;           db 0
178                              <1>
179                              <1> Dir_Str_Header:
180 000130B5 4469726563746F7279- <1>                 db "Directory: "
180 000130BE 3A20                <1>
181 000130C0 2F                  <1> Dir_Str_Root:   db "/"
182 000130C1 00<rep 40h>         <1> Dir_Str:        times 64 db 0
183 00013101 00000000            <1>                 dd 0
184 00013105 00                  <1>                 db 0
185                              <1>
186                              <1> Msg_Bad_Command:
187 00013106 42616420636F6D6D61- <1>                 db "Bad command or file name!"
187 0001310F 6E64206F722066696C- <1>
187 00013118 65206E616D6521      <1>
188 0001311F 0D0A00              <1>                 db 0Dh, 0Ah, 0
```

```
                                <1>
                                <1> msgl_drv_not_ready:
00013122 070D0A                 <1>             db 07h, 0Dh, 0Ah
                                <1>
                                <1> ; CMD_INTR.ASM - 09/11/2011 - Messages
                                <1>
                                <1> Msg_Not_Ready_Read_Err:
00013125 4472697665206E6F74-    <1>                 db "Drive not ready or read error!"
0001312E 207265616479206F72-    <1>
00013137 207265616420657272-    <1>
00013140 6F7221                 <1>
00013143 0D0A00                 <1>                 db 0Dh, 0Ah, 0
                                <1>
                                <1> Msg_Not_Ready_Write_Err:
00013146 4472697665206E6F74-    <1>                 db "Drive not ready or write error!"
0001314F 207265616479206F72-    <1>
00013158 207772697465206572-    <1>
00013161 726F7221               <1>
00013165 0D0A00                 <1>                 db 0Dh, 0Ah, 0
                                <1>
                                <1> Msg_Dir_Not_Found:
00013168 4469726563746F7279-    <1>                 db "Directory not found!"
00013171 206E6F7420666F756E-    <1>
0001317A 6421                   <1>
0001317C 0D0A00                 <1>                 db 0Dh, 0Ah, 0
                                <1>
                                <1> Msg_File_Not_Found:
0001317F 46696C65206E6F7420-    <1>                 db "File not found!"
00013188 666F756E6421           <1>
0001318E 0D0A00                 <1>                 db 0Dh, 0Ah, 0
                                <1>
                                <1> Msg_File_Directory_Not_Found:
00013191 46696C65206F722064-    <1>                 db "File or directory not found!"
0001319A 69726563746F727920-    <1>
000131A3 6E6F7420666F756E64-    <1>
000131AC 21                     <1>
000131AD 0D0A00                 <1>                 db 0Dh, 0Ah, 0
                                <1>
                                <1> Msg_LongName_Not_Found:
000131B0 4C6F6E67206E616D65-    <1>                 db "Long name not found!"
000131B9 206E6F7420666F756E-    <1>
000131C2 6421                   <1>
000131C4 0D0A00                 <1>                 db 0Dh, 0Ah, 0
                                <1>
                                <1> beep_Insufficient_Memory: ; 20/02/2017
000131C7 0D0A                   <1>             db 0Dh, 0Ah
000131C9 07                     <1>             db 07h
                                <1> Msg_Insufficient_Memory:
000131CA 496E73756666696369-    <1>                 db "Insufficient memory!"
000131D3 656E74206D656D6F72-    <1>
000131DC 7921                   <1>
000131DE 0D0A00                 <1>                 db 0Dh, 0Ah, 0
                                <1>
                                <1> Msg_Error_Code:
000131E1 436F6D6D616E642066-    <1>                 db 'Command failed! Error code : '
000131EA 61696C656421204572-    <1>
000131F3 726F7220636F646520-    <1>
000131FC 3A20                   <1>
000131FE 303068                 <1> error_code_hex: db '00h'
00013201 0D0A00                 <1>                 db 0Dh, 0Ah, 0
                                <1>
                                <1> ; 19/12/2025
                                <1> ;align 2
                                <1>
                                <1> ; 10/02/2016
                                <1> ; DIR.ASM - 09/10/2011
                                <1>
00013204 3C4449523E20202020-    <1> Type_Dir:       db '<DIR>     ' ; 10 bytes
0001320D 20                     <1>
                                <1>
                                <1> File_Name:
0001320E 20<rep Ch>             <1>                 times 12 db 20h
0001321A 20                     <1>             db 20h
                                <1> Dir_Or_FileSize:
0001321B 20<rep Ah>             <1>                 times 10 db 20h
00013225 20                     <1>             db 20h
                                <1> File_Attribute:
00013226 20202020               <1>             dd 20202020h
0001322A 20                     <1>             db 20h
                                <1> File_Day:
0001322B 3030                   <1>                 db '0','0'
0001322D 2F                     <1>             db '/'
                                <1> File_Month:
0001322E 3030                   <1>                 db '0','0'
00013230 2F                     <1>             db '/'
                                <1> File_Year:
00013231 30303030               <1>                 db '0','0','0','0'
00013235 20                     <1>             db 20h
                                <1> File_Hour:
00013236 3030                   <1>                 db '0','0'
00013238 3A                     <1>             db ':'
                                <1> File_Minute:
00013239 3030                   <1>                 db '0','0'
0001323B 00                     <1>             db 0
                                <1>
                                <1> Decimal_File_Count_Header:
0001323C 0D0A                   <1>             db 0Dh, 0Ah
                                <1> Decimal_File_Count:
0001323E 00<rep 6h>             <1>             times 6 db 0
                                <1>
00013244 2066696C6528732920-    <1> str_files:  db " file(s) & "
0001324D 2620                   <1>
                                <1> Decimal_Dir_Count:
0001324F 00<rep 6h>             <1>             times 6 db 0
                                <1> str_dirs:
00013255 206469726563746F72-    <1>             db " directory(s) "
0001325E 7928732920             <1>
00013263 0D0A00                 <1>             db 0Dh, 0Ah, 0
                                <1>
00013266 206279746528732920-    <1> str_bytes:  db " byte(s) in file(s)"
00013278 29                     <1>
00013279 0D0A00                 <1>             db 0Dh, 0Ah, 0
                                <1>
                                <1> ; CMD_INTR.ASM - 09/11/2011
                                <1> ; 07/10/2010
                                <1> Msg_invalid_name_chars:
0001327C 496E76616C69642066-    <1>                 db "Invalid file or directory name characters!"
00013285 696C65206F72206469-    <1>
0001328E 726563746F7279206E-    <1>
00013297 616D65206368617261-    <1>
000132A0 637465727321           <1>
000132A6 0D0A00                 <1>             db 0Dh, 0Ah, 0
                                <1> ; 21/02/2016
```

```
285 000132A9 46696C65206F722064- <1> Msg_Name_Exists: db "File or directory name exists!"
285 000132B2 69726563746F727920- <1>
285 000132BB 6E616D652065786973- <1>
285 000132C4 747321              <1>
286 000132C7 0D0A00              <1>                 db 0Dh, 0Ah, 0
287                              <1> Msg_DoYouWantMkdir:
288 000132CA 446F20796F75207761- <1>                 db "Do you want to make directory ", 0
288 000132D3 6E7420746F206D616B- <1>
288 000132DC 65206469726563746F- <1>
288 000132E5 72792000            <1>
289 000132E9 2028592F4E29203F20- <1> Msg_YesNo:      db " (Y/N) ? ", 0
289 000132F2 00                  <1>
290 000132F3 000D0A00            <1> Y_N_nextline:      db 0, 0Dh, 0Ah, 0
291 000132F7 4F4B2E0D0A00        <1> Msg_OK:            db "OK.", 0Dh, 0Ah, 0
292                              <1>
293                              <1> ; 27/02/2016
294                              <1> Msg_DoYouWantRmDir:
295 000132FD 446F20796F75207761- <1>                 db "Do you want to delete directory ", 0
295 00013306 6E7420746F2064656C- <1>
295 0001330F 657465206469726563- <1>
295 00013318 746F72792000        <1>
296                              <1> Msg_Dir_Not_Empty:
297 0001331E 4469726563746F7279- <1>                 db "Directory not empty!"
297 00013327 206E6F7420656D7074- <1>
297 00013330 7921                <1>
298 00013332 0D0A00              <1>                 db 0Dh, 0Ah, 0
299                              <1>
300                              <1> Msg_DoYouWantDelete:
301 00013335 446F20796F75207761- <1>                 db "Do you want to delete file ",0
301 0001333E 6E7420746F2064656C- <1>
301 00013347 6574652066696C6520- <1>
301 00013350 00                  <1>
302                              <1>
303 00013351 44656C657465642E2E- <1> Msg_Deleted:    db "Deleted...", 0Dh, 0Ah, 0
303 0001335A 2E0D0A00            <1>
304                              <1>
305                              <1> Msg_Permission_Denied:
306 0001335E 07                  <1>                 db 7
307 0001335F 5065726D697373696F- <1>                 db "Permission denied!", 0Dh, 0Ah, 0
307 00013368 6E2064656E69656421- <1>
307 00013371 0D0A00              <1>
308                              <1>
309                              <1> ; 04/03/2016
310 00013374 4E657720            <1> Msg_New:         db "New "
311 00013378 00                  <1>                 db 0
312                              <1> Str_Attributes:
313 00013379 417474726962757465- <1>                 db "Attributes : "
313 00013382 73203A20            <1>
314 00013386 4E4F524D414C        <1> Attr_Chars:      db "NORMAL"
315 0001338C 00                  <1>                 db 0
316                              <1>
317                              <1> ; 06/03/2016
318                              <1> ; CMD_INTR.ASM - 16/11/2010
319                              <1> Msg_DoYouWantRename:
320 0001338D 446F20796F75207761- <1>                 db "Do you want to rename ", 0
320 00013396 6E7420746F2072656E- <1>
320 0001339F 616D652000          <1>
321 000133A4 66696C652000        <1> Rename_File:     db "file ", 0
322 000133AA 6469726563746F7279- <1> Rename_Directory: db "directory ", 0
322 000133B3 2000                <1>
323 000133B5 00<rep Dh>          <1> Rename_OldName: times 13 db 0
324 000133C2 20617320            <1> Msg_File_rename_as: db " as "
325 000133C6 00<rep Dh>          <1> Rename_NewName: times 13 db 0
326                              <1>
327                              <1> ; 08/03/2016
328                              <1> ; CMD_INTR.ASM - 01/08/2010 - 23/04/2011
329                              <1> msg_not_same_drv:
330 000133D3 4E6F742073616D6520- <1>                 db "Not same drive!"
330 000133DC 647269766521        <1>
331 000133E2 0D0A00              <1>                 db 0Dh, 0Ah, 0
332                              <1>
333                              <1> Msg_DoYouWantMoveFile:
334 000133E5 446F20796F75207761- <1>                 db "Do you want to move file", 0
334 000133EE 6E7420746F206D6F76- <1>
334 000133F7 652066696C6500      <1>
335                              <1>
336                              <1> msg_insufficient_disk_space:
337 000133FE 496E73756666696369- <1>                 db "Insufficient disk space!"
337 00013407 656E74206469736B20- <1>
337 00013410 737061636521        <1>
338 00013416 0D0A00              <1>                 db 0Dh, 0Ah, 0
339                              <1>
340                              <1> ; 01/08/2010
341                              <1> msg_source_file:
342 00013419 0D0A536F7572636520- <1>             db 0Dh, 0Ah, "Source file name      :   "
342 00013422 66696C65206E616D65- <1>
342 0001342B 2020202020203A2020- <1>
342 00013434 20                  <1>
343                              <1> msg_source_file_drv:
344 00013435 203A00              <1>             db " :", 0
345                              <1> msg_destination_file:
346 00013438 0D0A44657374696E61- <1>             db 0Dh, 0Ah, "Destination file name :   "
346 00013441 74696F6E2066696C65- <1>
346 0001344A 206E616D65203A2020- <1>
346 00013453 20                  <1>
347                              <1> msg_destination_file_drv:
348 00013454 203A00              <1>             db " :", 0
349                              <1> msg_copy_nextline:
350 00013457 0D0A00              <1>             db 0Dh, 0Ah, 0
351                              <1>
352                              <1> ; 15/03/2016
353                              <1> ; CMD_INTR.ASM
354                              <1>
355                              <1> Msg_DoYouWantOverWriteFile:
356 0001345A 446F20796F75207761- <1>                 db "Do you want to overwrite file ",0
356 00013463 6E7420746F206F7665- <1>
356 0001346C 727772697465206669- <1>
356 00013475 6C652000            <1>
357                              <1>
358                              <1> Msg_DoYouWantCopyFile:
359 00013479 446F20796F75207761- <1>                 db "Do you want to copy file",0
359 00013482 6E7420746F20636F70- <1>
359 0001348B 792066696C6500      <1>
360                              <1>
361                              <1> Msg_read_file_error_before_EOF:
362 00013492 46696C652072656164- <1>             db "File reading error! (before EOF)"
362 0001349B 696E67206572726F72- <1>
362 000134A4 2120286265666F7265- <1>
362 000134AD 20454F4629          <1>
363 000134B2 0A0A00              <1>             db 0Ah, 0Ah, 0
364                              <1>
365                              <1> ; 18/03/2016
366                              <1> ; TRDOS 386 (v2.0) mainprog copy procedure
367                              <1> msg_reading:
```

```
368 000134B5 52656164696E672E2E- <1>            db "Reading... ", 0
368 000134BE 2E2000              <1>
369                              <1> msg_writing:
370 000134C1 57726974696E672E2E- <1>            db "Writing... ", 0
370 000134CA 2E2000              <1>
371                              <1> percentagestr:
372 000134CD 2020202500          <1>            db "   %", 0  ; "  0%" .. "100%"
373                              <1> ; 11/04/2016
374                              <1> Msg_No_Set_Space:
375 000134D2 496E73756666696369- <1>                db "Insufficient environment space!"
375 000134DB 656E7420656E766972- <1>
375 000134E4 6F6E6D656E74207370- <1>
375 000134ED 61636521            <1>
376 000134F1 0D0A00              <1>                db 0Dh, 0Ah, 0
377                              <1> ; 18/04/2016
378                              <1> isc_msg:
379 000134F4 0D0A                <1>            db 0Dh, 0Ah
380 000134F6 494E56414C49442053- <1>            db "INVALID SYSTEM CALL", 0
380 000134FF 595354454D2043414C- <1>
380 00013508 4C00                <1>
381                              <1> usi_msg:
382 0001350A 0D0A                <1>            db 0Dh, 0Ah
383 0001350C 554E444546494E4544- <1>            db "UNDEFINED SOFTWARE INTERRUPT", 0
383 00013515 20534F465457415245- <1>
383 0001351E 20494E544552525550- <1>
383 00013527 5400                <1>
384                              <1> ifc_msg:
385 00013529 0D0A                <1>            db 0Dh, 0Ah
386 0001352B 494E56414C49442046- <1>            db "INVALID FUNCTION CALL"
386 00013534 554E4354494F4E2043- <1>
386 0001353D 414C4C              <1>
387                              <1> inv_msg_for_trdos_v2:
388 00013540 20                  <1>            db 20h
389 00013541 666F72205452444F53- <1>            db "for TRDOS v2!"
389 0001354A 20763221            <1>
390 0001354E 07                  <1>            db 07h
391 0001354F 0D0A                <1>            db 0Dh, 0Ah
392 00013551 0D0A                <1>            db 0Dh, 0Ah
393 00013553 494E5420            <1>            db "INT "
394 00013557 303068              <1> int_num_str:      db "00h"
395 0001355A 0D0A                <1>            db 0Dh, 0Ah
396 0001355C 454158203A20        <1>            db "EAX : "
397 00013562 303030303030303068- <1> eax_str:    db "00000000h", 0Dh, 0Ah
397 0001356B 0D0A                <1>
398 0001356D 454950203A20        <1>            db "EIP : "
399 00013573 303030303030303068- <1> eip_str:    db "00000000h", 0Dh, 0Ah, 0
399 0001357C 0D0A00              <1>
400                              <1>
401                              <1> ; 17/04/2021
402                              <1> ; ('KDEV' device parameters are disabled as temporary)
403                              <1>
404                              <1> ; 07/10/2016
405                              <1> ; Device names & parameters (for kernel devices)
406                              <1>
407 0001357F 90                  <1> align 2
408                              <1> ;KDEV_NAME:
409                              <1> ;           db 'TTY',0,0,0,0,0 ; 1
410                              <1> ;           db 'MEM',0,0,0,0,0 ; 2
411                              <1> ;           db 'FD0',0,0,0,0,0 ; 3
412                              <1> ;           db 'FD1',0,0,0,0,0 ; 4
413                              <1> ;           db 'HD0',0,0,0,0,0 ; 5
414                              <1> ;           db 'HD1',0,0,0,0,0 ; 6
415                              <1> ;           db 'HD2',0,0,0,0,0 ; 7
416                              <1> ;           db 'HD3',0,0,0,0,0 ; 8
417                              <1> ;           db 'LPT',0,0,0,0,0 ; 9
418                              <1> ;           db 'TTY0',0,0,0,0 ; 10
419                              <1> ;           db 'TTY1',0,0,0,0 ; 11
420                              <1> ;           db 'TTY2',0,0,0,0 ; 12
421                              <1> ;           db 'TTY3',0,0,0,0 ; 13
422                              <1> ;           db 'TTY4',0,0,0,0 ; 14
423                              <1> ;           db 'TTY5',0,0,0,0 ; 15
424                              <1> ;           db 'TTY6',0,0,0,0 ; 16
425                              <1> ;           db 'TTY7',0,0,0,0 ; 17
426                              <1> ;           db 'TTY8',0,0,0,0 ; 18
427                              <1> ;           db 'TTY9',0,0,0,0 ; 19
428                              <1> ;           db 'COM1',0,0,0,0 ; 18
429                              <1> ;           db 'COM2',0,0,0,0 ; 19
430                              <1> ;           ;db 'CONSOLE',0          ; 1
431                              <1> ;           ;db 'PRINTER',0   ; 9
432                              <1> ;           ;db 'CDROM'      ; 20
433                              <1> ;           ;db 'CDROM0'     ; 20
434                              <1> ;           ;db 'CDROM1'     ; 21
435                              <1> ;           ;db 'DVD'        ; 22
436                              <1> ;           ;db 'DVD0'       ; 22
437                              <1> ;           ;db 'DVD1'       ; 23
438                              <1> ;           ;db 'USB'        ; 24
439                              <1> ;           ;db 'USB0'       ; 24
440                              <1> ;           ;db 'USB1'       ; 25
441                              <1> ;           ;db 'USB2'       ; 26
442                              <1> ;           ;db 'USB3'        ; 27
443                              <1> ;           ;db 'KEYBOARD'   ; 1
444                              <1> ;           ;db 'MOUSE'      ; 28
445                              <1> ;           ;db 'SOUND'      ; 29
446                              <1> ;           ;db 'VGA',0,0,0,0 ; 30
447                              <1> ;           ;db 'CGA',0,0,0,0 ; 31
448                              <1> ;           ;db 'AUDIO',0,0,0 ; 29
449                              <1> ;           ;db 'VIDEO',0,0,0 ; 32
450                              <1> ;           ;db 'MUSIC',0,0,0 ; 33
451                              <1> ;           ;db 'ETHERNET'   ; 34
452                              <1> ;           ;db 'SD0',0,0,0,0,0 ; 35
453                              <1> ;           ;db 'SD1',0,0,0,0,0 ; 36
454                              <1> ;           ;db 'SD2',0,0,0,0,0 ; 37
455                              <1> ;           ;db 'SD3',0,0,0,0,0 ; 38
456                              <1> ;           ;db 'SATA0'      ; 35
457                              <1> ;           ;db 'SATA1'      ; 36
458                              <1> ;           ;db 'SATA2'        ; 37
459                              <1> ;           ;db 'SATA3'        ; 38
460                              <1> ;           ;db 'PATA0',0,0,0  ; 5
461                              <1> ;           ;db 'PATA1',0,0,0  ; 6
462                              <1> ;           ;db 'PATA2',0,0,0  ; 7
463                              <1> ;           ;db 'PATA3',0,0,0  ; 8
464                              <1> ;           ;db 'WIRELESS'   ; 39
465                              <1> ;           ;db 'HDMI',0,0,0,0 ; 40
466                              <1> ;           db 'NULL',0,0,0,0 ; 0
467                              <1>
468                              <1> ;NumOfKernelDevNames equ ($-KDEV_NAME) / 8 ; 20 (07/10/2016)
469                              <1>
470                              <1> ;KDEV_NUMBER:
471                              <1> ;           db 1,2,3,4,5,6,7,8,9
472                              <1> ;           db 10,11,12,13,14,15,16,17,18,19
473                              <1> ;           db 18,19,0
474                              <1>
475                              <1> ;NumOfKernelDevices equ $ - KDEV_NUMBER
476                              <1>
```

```
                        <1> ;KDEV_OADDR:
                        <1> ;           dd otty ;tty  ; 1
                        <1> ;           dd sret ;mem  ; 2
                        <1> ;           dd sret ;fd0  ; 3
                        <1> ;           dd sret ;fd1  ; 4
                        <1> ;           dd sret ;hd0  ; 5
                        <1> ;           dd sret ;hd1  ; 6
                        <1> ;           dd sret ;hd2  ; 7
                        <1> ;           dd sret ;hd3  ; 8
                        <1> ;           dd sret ;lpt  ; 9
                        <1> ;           dd ocvt ;tty0 ; 10
                        <1> ;           dd ocvt ;tty1 ; 11
                        <1> ;           dd ocvt ;tty2 ; 12
                        <1> ;           dd ocvt ;tty3 ; 13
                        <1> ;           dd ocvt ;tty4 ; 14
                        <1> ;           dd ocvt ;tty5 ; 15
                        <1> ;           dd ocvt ;tty6 ; 16
                        <1> ;           dd ocvt ;tty7 ; 17
                        <1> ;           dd ocvt ;tty8 ; 18
                        <1> ;           dd ocvt ;tty9 ; 19
                        <1> ;           ;dd ocvt ;com1 ; 18
                        <1> ;           ;dd ocvt ;com2 ; 19
                        <1> ;           dd sret ;null ; 20
                        <1> ;KDEV_CADDR:
                        <1> ;           dd ctty ;tty  ; 1
                        <1> ;           dd cret ;mem  ; 2
                        <1> ;           dd cret ;fd0  ; 3
                        <1> ;           dd cret ;fd1  ; 4
                        <1> ;           dd cret ;hd0  ; 5
                        <1> ;           dd cret ;hd1  ; 6
                        <1> ;           dd cret ;hd2  ; 7
                        <1> ;           dd cret ;hd3  ; 8
                        <1> ;           dd cret ;lpt  ; 9
                        <1> ;           dd ocvt ;tty0 ; 10
                        <1> ;           dd ccvt ;tty1 ; 11
                        <1> ;           dd ccvt ;tty2 ; 12
                        <1> ;           dd ccvt ;tty3 ; 13
                        <1> ;           dd ccvt ;tty4 ; 14
                        <1> ;           dd ccvt ;tty5 ; 15
                        <1> ;           dd ccvt ;tty6 ; 16
                        <1> ;           dd ccvt ;tty7 ; 17
                        <1> ;           dd ccvt ;tty8 ; 18
                        <1> ;           dd ccvt ;tty9 ; 19
                        <1> ;           ;dd ccvt ;com1 ; 18
                        <1> ;           ;dd ccvt ;com2 ; 19
                        <1> ;           dd cret ;null ; 20
                        <1> ;
                        <1> ;KDEV_RADDR:
                        <1> ;           dd rtty ;tty  ; 1
                        <1> ;           dd rmem ;mem  ; 2
                        <1> ;           dd rfd  ;fd0  ; 3
                        <1> ;           dd rfd  ;fd1  ; 4
                        <1> ;           dd rhd  ;hd0  ; 5
                        <1> ;           dd rhd  ;hd1  ; 6
                        <1> ;           dd rhd  ;hd2  ; 7
                        <1> ;           dd rhd  ;hd3  ; 8
                        <1> ;           dd rlpt ;lpt  ; 9
                        <1> ;           dd rcvt ;tty0 ; 10
                        <1> ;           dd rcvt ;tty1 ; 11
                        <1> ;           dd rcvt ;tty2 ; 12
                        <1> ;           dd rcvt ;tty3 ; 13
                        <1> ;           dd rcvt ;tty4 ; 14
                        <1> ;           dd rcvt ;tty5 ; 15
                        <1> ;           dd rcvt ;tty6 ; 16
                        <1> ;           dd rcvt ;tty7 ; 17
                        <1> ;           dd rcvt ;tty8 ; 18
                        <1> ;           dd rcvt ;tty9 ; 19
                        <1> ;           ;dd rcvt ;com1 ; 18
                        <1> ;           ;dd rcvt ;com2 ; 19
                        <1> ;           dd rnull ;null ; 20
                        <1> ;KDEV_WADDR:
                        <1> ;           dd wtty ;tty  ; 1
                        <1> ;           dd wmem ;mem  ; 2
                        <1> ;           dd wfd  ;fd0  ; 3
                        <1> ;           dd wfd  ;fd1  ; 4
                        <1> ;           dd whd  ;hd0  ; 5
                        <1> ;           dd whd  ;hd1  ; 6
                        <1> ;           dd whd  ;hd2  ; 7
                        <1> ;           dd whd  ;hd3  ; 8
                        <1> ;           dd wlpt ;lpt  ; 9
                        <1> ;           dd xmtt ;tty0 ; 10
                        <1> ;           dd xmtt ;tty1 ; 11
                        <1> ;           dd xmtt ;tty2 ; 12
                        <1> ;           dd xmtt ;tty3 ; 13
                        <1> ;           dd xmtt ;tty4 ; 14
                        <1> ;           dd xmtt ;tty5 ; 15
                        <1> ;           dd xmtt ;tty6 ; 16
                        <1> ;           dd xmtt ;tty7 ; 17
                        <1> ;           dd xmtt ;tty8 ; 18
                        <1> ;           dd xmtt ;tty9 ; 19
                        <1> ;           ;dd xmtt ;com1 ; 18
                        <1> ;           ;dd xmtt ;com2 ; 19
                        <1> ;           dd wnull ;null ; 20
                        <1>
                        <1> ; DEV_ACCESS bits:
                        <1>     ; 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
                        <1>
                        <1> ;KDEV_ACCESS: ; 08/10/2016
                        <1> ;           db  00000111b  ; tty, 1
                        <1> ;           db  00000111b  ; mem, 2
                        <1> ;           db  10001111b  ; fd0, 3
                        <1> ;           db  10001111b  ; fd1, 4
                        <1> ;           db  10001111b  ; hd0, 5
                        <1> ;           db  10001111b  ; hd1, 6
                        <1> ;           db  10001111b  ; hd2, 7
                        <1> ;           db  10001111b  ; hd3, 8
                        <1> ;           db  00000111b   ; lpt, 9
                        <1> ;           db  00000111b  ; tty0, 10
                        <1> ;           db  00000111b  ; tty1, 11
                        <1> ;           db  00000111b  ; tty2, 12
                        <1> ;           db  00000111b  ; tty3, 13
                        <1> ;           db  00000111b  ; tty4, 14
                        <1> ;           db  00000111b  ; tty5, 15
                        <1> ;           db  00000111b  ; tty6, 16
                        <1> ;           db  00000111b  ; tty7, 17
                        <1> ;           db  00000111b  ; tty8, 18
                        <1> ;           db  00000111b  ; tty9, 19
```

```
                            <1> ;           ;db 00000111b  ; com1, 18
                            <1> ;           ;db 00000111b  ; com2, 19
                            <1> ;           db  00000000b   ; null, 0
                            <1>
                            <1> ; 07/10/2016
                            <1> ;NumOfInstallableDevices equ 8
                            <1> ;NUMIDEV    equ NumOfInstallableDevices ; 8
                            <1> ;NUMOFDEVICES      equ NumOfKernelDevices + NumOfInstallableDevices
                            <1>
                            <1> ; 26/02/2017
                            <1> ; IRQ Callback (& Signal Response Byte) service availibity
                            <1> ; 'syscalbac'
                            <1> ; ************************************************
                            <1> ; IRQ 00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
                            <1> ; --- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
                            <1> ; --- 00 00 00 01 02 03 00 04 00 05 06 07 08 09 00 00
                            <1> ; ************************************************
                            <1> IRQenum:
00013580 00000010203000400- <1>     db  0,0,0,1,2,3,0,4,0,5,6,7,8,9,0,0
00013589 05060708090000     <1>
                            <1>
                            <1> ; 28/08/2017
                            <1> ; 20/08/2017
                            <1> ; DMA Registers (for 'sysdma')
                            <1> ; 02/07/2017 (sb16mod.s)
00013590 00020406C0C4C8CC   <1> dma_adr:    db 0,2,4,6,0C0h,0C4h,0C8h,0CCh
00013598 01030507C2C6CACE   <1> dma_cnt:    db 1,3,5,7,0C2h,0C6h,0CAh,0CEh
000135A0 878381828F8B898A   <1> dma_page:   db 87h,83h,81h,82h,8Fh,8Bh,89h,8Ah ; 03/08/2017
000135A8 0A0A0A0AD4D4D4D4   <1> dma_mask:   db 0Ah,0Ah,0Ah,0Ah,0D4h,0D4h,0D4h,0D4h
000135B0 0B0B0B0BD6D6D6D6   <1> dma_mod:    db 0Bh,0Bh,0Bh,0Bh,0D6h,0D6h,0D6h,0D6h
000135B8 0C0C0C0CD8D8D8D8   <1> dma_flip:   db 0Ch,0Ch,0Ch,0Ch,0D8h,0D8h,0D8h,0D8h
                            3441
                            3442                                ; 27/08/2014
                            3443                                scr_row:
000135C0 E0810B00           3444                                    dd 0B8000h + 0A0h + 0A0h + 0A0h ; Row 3
                            3445                                scr_col:
000135C4 00000000           3446                                    dd 0
                            3447
                            3448                                Align 4
                            3449                                    ; 15/04/2016
                            3450                                    ; TRDOS 386 (TRDOS v2.0)
                            3451
                            3452                                    ; 21/08/2014
                            3453                                ilist:
                            3454                                    ;times  32 dd cpu_except ; INT 0 to INT 1Fh
                            3455                                    ;
                            3456                                    ; Exception list
                            3457                                    ; 25/08/2014
000135C8 [1E0C0000]         3458                                    dd      exc0    ; 0h,  Divide-by-zero Error
000135CC [250C0000]         3459                                    dd      exc1
000135D0 [2C0C0000]         3460                                    dd      exc2
000135D4 [330C0000]         3461                                    dd      exc3
000135D8 [370C0000]         3462                                    dd      exc4
000135DC [3B0C0000]         3463                                    dd      exc5
000135E0 [3F0C0000]         3464                                    dd      exc6    ; 06h,  Invalid Opcode
000135E4 [430C0000]         3465                                    dd      exc7
000135E8 [470C0000]         3466                                    dd      exc8
000135EC [4B0C0000]         3467                                    dd      exc9
000135F0 [4F0C0000]         3468                                    dd      exc10
000135F4 [530C0000]         3469                                    dd      exc11
000135F8 [570C0000]         3470                                    dd      exc12
000135FC [5B0C0000]         3471                                    dd      exc13   ; 0Dh, General Protection Fault
00013600 [5F0C0000]         3472                                    dd      exc14   ; 0Eh, Page Fault
00013604 [630C0000]         3473                                    dd      exc15
00013608 [670C0000]         3474                                    dd      exc16
0001360C [6B0C0000]         3475                                    dd      exc17
00013610 [6F0C0000]         3476                                    dd      exc18
00013614 [730C0000]         3477                                    dd      exc19
00013618 [770C0000]         3478                                    dd      exc20
0001361C [7B0C0000]         3479                                    dd      exc21
00013620 [7F0C0000]         3480                                    dd      exc22
00013624 [830C0000]         3481                                    dd      exc23
00013628 [870C0000]         3482                                    dd      exc24
0001362C [8B0C0000]         3483                                    dd      exc25
00013630 [8F0C0000]         3484                                    dd      exc26
00013634 [930C0000]         3485                                    dd      exc27
00013638 [970C0000]         3486                                    dd      exc28
0001363C [9B0C0000]         3487                                    dd      exc29
00013640 [9F0C0000]         3488                                    dd      exc30
00013644 [A30C0000]         3489                                    dd      exc31
                            3490                                IRQ_list: ; 28/02/2017 ('syscalbac')
                            3491                                    ; Interrupt list
00013648 [88090000]         3492                                    dd      timer_int       ; INT 20h
                            3493                                            ;dd     irq0
0001364C [ED100000]         3494                                    dd      kb_int          ; 24/01/2016
                            3495                                            ;dd     irq1
00013650 [740B0000]         3496                                    dd      irq2
                            3497                                            ; COM2 int
00013654 [780B0000]         3498                                    dd      irq3
                            3499                                            ; COM1 int
00013658 [830B0000]         3500                                    dd      irq4
0001365C [8E0B0000]         3501                                    dd      irq5
                            3502                                ;DISKETTE_INT: ;06/02/2015
00013660 [A04E0000]         3503                                    dd      fdc_int         ; 16/02/2015, IRQ 6 handler
                            3504                                            ;dd     irq6
                            3505                                ; Default IRQ 7 handler against spurious IRQs (from master PIC)
                            3506                                ; 25/02/2015 (source: http://wiki.osdev.org/8259_PIC)
00013664 [FA0E0000]         3507                                    dd      default_irq7    ; 25/02/2015
                            3508                                            ;dd     irq7
                            3509                                ; Real Time Clock Interrupt
00013668 [FC0A0000]         3510                                    dd      rtc_int         ; 23/02/2015, IRQ 8 handler
                            3511                                            ;dd     irq8    ; INT 28h
0001366C [9E0B0000]         3512                                    dd      irq9
00013670 [A20B0000]         3513                                    dd      irq10
00013674 [A60B0000]         3514                                    dd      irq11
00013678 [AA0B0000]         3515                                    dd      irq12
0001367C [AE0B0000]         3516                                    dd      irq13
                            3517                                ;HDISK_INT1:  ;06/02/2015
00013680 [DF570000]         3518                                    dd      hdc1_int        ; 21/02/2015, IRQ 14 handler
                            3519                                            ;dd     irq14
                            3520                                ;HDISK_INT2:  ;06/02/2015
00013684 [02580000]         3521                                    dd      hdc2_int        ; 21/02/2015, IRQ 15 handler
                            3522                                            ;dd     irq15   ; INT 2Fh
                            3523                                            ; 14/08/2015
                            3524                                    ;dd     sysent          ; INT 30h (system calls)
                            3525
                            3526                                    ; 15/04/2016
                            3527                                    ; TRDOS 386(TRDOS v2.0) Software Interrupts
                            3528
00013688 [E5360100]         3529                                    dd      int30h          ; Reserved for
                            3530                                                            ; !!! Retro UNIX (RUNIX) !!!
                            3531                                                            ; !!! SINGLIX !!! System Calls
0001368C [7C170000]         3532                                    dd      int31h          ; Video BIOS (IBM PC/AT, Int 10h)
00013690 [490F0000]         3533                                    dd      int32h          ; Keyboard Functions (IBM PC/AT, Int 16h)
```

```
3534 00013694 [424F0000]                dd      int33h          ; DISK I/O (IBM PC/AT, Int 13h)
3535 00013698 [6B1B0100]                dd      int34h          ; #IOCTL# (I/O port access support for ring 3)
3536 0001369C [E5620000]                dd      int35h          ; Time/Date Functions (IBM PC/AT, Int 1Ah)
3537 000136A0 [AE0D0000]                dd      ignore_int      ; INT 36h : Timer Functions
3538 000136A4 [AE0D0000]                dd      ignore_int      ; INT 37h
3539 000136A8 [AE0D0000]                dd      ignore_int      ; INT 38h
3540 000136AC [AE0D0000]                dd      ignore_int      ; INT 39h
3541 000136B0 [AE0D0000]                dd      ignore_int      ; INT 3Ah
3542 000136B4 [AE0D0000]                dd      ignore_int      ; INT 3Bh
3543 000136B8 [AE0D0000]                dd      ignore_int      ; INT 3Ch
3544 000136BC [AE0D0000]                dd      ignore_int      ; INT 3Dh
3545 000136C0 [AE0D0000]                dd      ignore_int      ; INT 3Eh
3546 000136C4 [AE0D0000]                dd      ignore_int      ; INT 3Fh
3547 000136C8 [C6C80000]                dd      sysent          ; INT 40h : !!! TRDOS 386 System Calls !!!
3548                                    ;dd     ignore_int
3549 000136CC 00000000                  dd      0
3550
3551                                ; 20/08/2014
3552                                  ; /* This is the default interrupt "handler" :-) */
3553                                  ; Linux v0.12 (head.s)
3554                                int_msg:
3555 000136D0 556E6B6E6F776E2069-       db "Unknown interrupt ! ", 0
3555 000136D9 6E7465727275707420-
3555 000136E2 212000
3556
3557                                ; 15/04/2016
3558                                ; TRDOS 386 (TRDOS v2.0)
3559
3560                                ; 29/04/2016
3561                                int30h:
3562                                trdos_isc_routine:
3563                                    ; 02/05/2016
3564                                    ; 01/05/2016
3565                                    ; 29/04/2016
3566                                    ; 18/04/2016
3567                                    ; 15/04/2016 (TRDOS 386 = TRDOS v2.0)
3568                                    ; 17/04/2011 (TRDOS v1.0, 'IFC.ASM')
3569                                    ; 03/02/2011 ('trdos_ifc_routine')
3570                                    ;
3571 000136E5 8B1C24                  mov     ebx, [esp] ; EIP (next)
3572 000136E8 83EB02                  sub     ebx, 2 ; EIP (CD ##h)
3573
3574 000136EB 89C1                    mov     ecx, eax
3575 000136ED 8A4301                  mov     al, [ebx+1] ; CDh ##h
3576
3577 000136F0 66BA1000                mov     dx, KDATA
3578 000136F4 8EDA                    mov     ds, dx
3579 000136F6 8EC2                    mov     es, dx
3580
3581 000136F8 FC                      cld
3582 000136F9 8B15[80760100]          mov     edx, [k_page_dir]
3583 000136FF 0F22DA                  mov     cr3, edx
3584
3585 00013702 E8B40AFFFF              call    bytetohex
3586 00013707 66A3[57350100]          mov     [int_num_str], ax
3587
3588 0001370D 89D8                    mov     eax, ebx ; EIP
3589 0001370F E8E70AFFFF              call    dwordtohex
3590 00013714 8915[73350100]          mov     [eip_str], edx
3591 0001371A A3[77350100]            mov     [eip_str+4], eax
3592
3593 0001371F 89C8                    mov     eax, ecx
3594 00013721 E8D50AFFFF              call    dwordtohex
3595 00013726 8915[62350100]          mov     [eax_str], edx
3596 0001372C A3[66350100]            mov     [eax_str+4], eax
3597
3598 00013731 43                      inc     ebx
3599 00013732 8A03                    mov     al, [ebx] ; Interrupt number
3600
3601                                trdos_isc_handler:
3602 00013734 80FE30                  cmp     dh, 30h ; Retro UNIX, SINGLIX System calls
3603 00013737 7507                    jne     short trdos_usi_handler
3604 00013739 BE[F4340100]            mov     esi, isc_msg
3605 0001373E EB05                    jmp     short loc_write_inv_system_call_msg
3606
3607                                trdos_usi_handler:
3608 00013740 BE[0A350100]            mov     esi, usi_msg
3609
3610                                loc_write_inv_system_call_msg:
3611 00013745 E86036FFFF              call    print_msg
3612                                    ; 29/04/2016
3613 0001374A BE[40350100]            mov     esi, inv_msg_for_trdos_v2
3614 0001374F E85636FFFF              call    print_msg
3615
3616                                loc_ifc_terminate_process:
3617                                    ; u.uno = process number
3618                                    ; 29/04/2016
3619
3620                                    ; 02/05/2016
3621 00013754 FE05[288E0100]            inc     byte [sysflg] ; 0FFh -> 0
3622
3623 0001375A B801000000                mov     eax, 1
3624 0001375F E95894FFFF                jmp     sysexit
3625
3626                                ; 07/03/2015
3627                                ; Temporary Code
3628                                display_disks:
3629 00013764 803D[7C660000]00          cmp     byte [fd0_type], 0
3630 0001376B 7605                      jna     short ddsks1
3631 0001376D E87D000000                call    pdskm
3632                                ddsks1:
3633 00013772 803D[7D660000]00          cmp     byte [fd1_type], 0
3634 00013779 760C                      jna     short ddsks2
3635 0001377B C605[BF380100]31          mov     byte [dskx], '1'
3636 00013782 E868000000                call    pdskm
3637                                ddsks2:
3638 00013787 803D[7E660000]00          cmp     byte [hd0_type], 0
3639 0001378E 7654                      jna     short ddsk6
3640 00013790 66C705[BD380100]68-       mov     word [dsktype], 'hd'
3640 00013798 64
3641 00013799 C605[BF380100]30          mov     byte [dskx], '0'
3642 000137A0 E84A000000                call    pdskm
3643                                ddsks3:
3644 000137A5 803D[7F660000]00          cmp     byte [hd1_type], 0
3645 000137AC 7636                      jna     short ddsk6
3646 000137AE C605[BF380100]31          mov     byte [dskx], '1'
3647 000137B5 E835000000                call    pdskm
3648                                ddsks4:
3649 000137BA 803D[80660000]00          cmp     byte [hd2_type], 0
3650 000137C1 7621                      jna     short ddsk6
3651 000137C3 C605[BF380100]32          mov     byte [dskx], '2'
3652 000137CA E820000000                call    pdskm
3653                                ddsks5:
3654 000137CF 803D[81660000]00          cmp     byte [hd3_type], 0
```

```
3655 000137D6 760C                      jna     short ddsk6
3656 000137D8 C605[BF380100]33          mov     byte [dskx], '3'
3657 000137DF E80B000000                call    pdskm
3658                                    ddsk6:
3659 000137E4 BE[E7380100]              mov     esi, nextline
3660 000137E9 E806000000                call    pdskml
3661                                    pdskm_ok:
3662 000137EE C3                        retn
3663                                    pdskm:
3664 000137EF BE[BB380100]              mov     esi, dsk_ready_msg
3665                                    pdskml:
3666 000137F4 AC                        lodsb
3667 000137F5 08C0                      or      al, al
3668 000137F7 74F5                      jz      short pdskm_ok
3669 000137F9 56                        push    esi
3670                                    ; 13/05/2016
3671 000137FA BB07000000                    mov     ebx, 7  ; Black background,
3672                                                        ; light gray forecolor
3673                                                        ; Video page 0 (bh=0)
3674 000137FF E8D7EAFEFF                call    _write_tty
3675 00013804 5E                        pop     esi
3676 00013805 EBED                      jmp     short pdskml
3677
3678 00013807 90                        Align 2
3679                                    ; 21/08/2014
3680                                    exc_msg:
3681 00013808 435055206578636570-           db "CPU exception ! "
3681 00013811 74696F6E202120
3682                                    excnstr:            ; 25/08/2014
3683 00013818 3F3F68202045495020-           db "??h", "  EIP : "
3683 00013821 3A20
3684                                    EIPstr: ; 29/08/2014
3685 00013823 00<rep Ch>                    times 12 db 0
3686
3687                                    ; 23/02/2015
3688                                    ; 25/08/2014
3689                                    ;scounter:
3690                                    ;   db 5
3691                                    ;   db 19
3692
3693                                    ; 06/11/2014
3694                                    ; Memory Information message
3695                                    ; 14/08/2015
3696                                    msg_memory_info:
3697 0001382F 07                            db      07h
3698 00013830 0D0A                          db      0Dh, 0Ah
3699                                        ;db     "MEMORY ALLOCATION INFO", 0Dh, 0Ah, 0Dh, 0Ah
3700 00013832 546F74616C206D656D-           db      "Total memory : "
3700 0001383B 6F7279203A20
3701                                    mem_total_b_str: ; 10 digits
3702 00013841 303030303030303030-           db      "0000000000 bytes", 0Dh, 0Ah
3702 0001384A 302062797465730D0A
3703 00013853 2020202020202020-             db      "               ", 20h, 20h, 20h
3703 0001385C 2020202020202020
3704                                    mem_total_p_str: ; 7 digits
3705 00013865 303030303030302070-           db      "0000000 pages", 0Dh, 0Ah
3705 0001386E 616765730D0A
3706 00013874 0D0A                          db      0Dh, 0Ah
3707 00013876 46726565206D656D6F-           db      "Free memory  : "
3707 0001387F 727920203A20
3708                                    free_mem_b_str:  ; 10 digits
3709 00013885 3F3F3F3F3F3F3F3F3F-           db      "?????????? bytes", 0Dh, 0Ah
3709 0001388E 3F2062797465730D0A
3710 00013897 2020202020202020-             db      "               ", 20h, 20h, 20h
3710 000138A0 2020202020202020
3711                                    free_mem_p_str:  ; 7 digits
3712 000138A9 3F3F3F3F3F3F3F2070-           db      "??????? pages", 0Dh, 0Ah
3712 000138B2 616765730D0A
3713 000138B8 0D0A00                        db      0Dh, 0Ah, 0
3714
3715                                    dsk_ready_msg:
3716 000138BB 0D0A                          db      0Dh, 0Ah
3717                                    dsktype:
3718 000138BD 6664                          db      'fd'
3719                                    dskx:
3720 000138BF 30                            db      '0'
3721 000138C0 20                            db      20h
3722 000138C1 697320524541445920-           db      'is READY ...'
3722 000138CA 2E2E2E
3723 000138CD 00                            db      0
3724
3725                                    setup_error_msg:
3726 000138CE 0D0A                          db      0Dh, 0Ah
3727 000138D0 4469736B2053657475-           db      'Disk Setup Error !'
3727 000138D9 70204572726F722021
3728 000138E2 0D0A00                        db      0Dh, 0Ah,0
3729
3730                                    next2line: ; 08/02/2016
3731 000138E5 0D0A                          db      0Dh, 0Ah
3732                                    nextline:
3733 000138E7 0D0A00                        db      0Dh, 0Ah, 0
3734
3735                                    ; temporary
3736                                    ; 19/12/2020
3737                                    msg_lfb_addr:
3738                                        ;db     0Dh, 0Ah
3739 000138EA 4C696E656172206672-           db      "Linear frame buffer at "
3739 000138F3 616D65206275666665-
3739 000138FC 7220617420
3740                                    lfb_addr_str: ; 8 (hex) digits
3741 00013901 303030303030303068-           db      "00000000h", 0Dh, 0Ah
3741 0001390A 0D0A
3742 0001390C 0D0A00                        db      0Dh, 0Ah, 0
3743
3744                                    ; KERNEL - SYSINIT Messages
3745                                    ; 24/08/2015
3746                                    ; 13/04/2015 - (Retro UNIX 386 v1 Beginning)
3747                                    ; 14/07/2013
3748                                    ;kernel_init_err_msg:
3749                                    ;   db 0Dh, 0Ah
3750                                    ;   db 07h
3751                                    ;   db 'Kernel initialization ERROR !'
3752                                    ;   db 0Dh, 0Ah, 0
3753
3754                                    ;welcome_msg:
3755                                    ;   db 0Dh, 0Ah
3756                                    ;   db 07h
3757                                    ;   db 'Welcome to TRDOS 386 Operating System !'
3758                                    ;   db 0Dh, 0Ah
3759                                    ;   db 'by Erdogan Tan - 31/12/2017 (v2.0.0)'
3760                                    ;   db 0Dh, 0Ah, 0
3761
3762                                    panic_msg:
3763 0001390F 0D0A07                        db 0Dh, 0Ah, 07h
```

```
3764 00013912 4552524F523A204B65-         db 'ERROR: Kernel Panic !'
3764 0001391B 726E656C2050616E69-
3764 00013924 632021
3765 00013927 0D0A00                      db 0Dh, 0Ah, 0
3766
3767                                  ;msgl_drv_not_ready:
3768                                  ;   db 07h, 0Dh, 0Ah
3769                                  ;       db 'Drive not ready or read error !'
3770                                  ;       db 0Dh, 0Ah, 0
3771
3772                                  starting_msg:
3773                                      ;;;;;;;db "Turkish Rational DOS v2.0 [18/04/2021] ...", 0
3774                                      ;;;;;;db "Turkish Rational DOS v2.0 [11/08/2022] ...", 0
3775                                      ;;;;;db "Turkish Rational DOS v2.0 [30/08/2023] ...", 0
3776                                      ;;;;db "Turkish Rational DOS v2.0 [07/12/2023] ...", 0
3777                                      ;;;db "Turkish Rational DOS v2.0 [23/06/2024] ...", 0
3778                                      ;;db "Turkish Rational DOS v2.0 [29/12/2024] ...", 0
3779                                      ;db "Turkish Rational DOS v2.0 [28/01/2025] ...", 0
3780 0001392A 5475726B6973682052-         db "Turkish Rational DOS v2.0 [28/12/2025] ...", 0
3780 00013933 6174696F6E616C2044-
3780 0001393C 4F532076322E30205B-
3780 00013945 32382F31322F323032-
3780 0001394E 355D202E2E2E00
3781
3782                                  NextLine:
3783 00013955 0D0A00                      db 0Dh, 0Ah, 0
3784
3785                                  %include 'audio.s' ; 03/04/2017
   1                                  <1> ; ****************************************************************************
   2                                  <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.10 - audio.s
   3                                  <1> ; ----------------------------------------------------------------------------
   4                                  <1> ; Last Update: 28/01/2025  (Previous: 06/06/2024 - Kernel v2.0.9)
   5                                  <1> ; ----------------------------------------------------------------------------
   6                                  <1> ; Beginning: 03/04/2017
   7                                  <1> ; ----------------------------------------------------------------------------
   8                                  <1> ; Assembler: NASM version 2.15 (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    03FFFFFFh ; 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
   101                                  <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
   132                                  <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> FLAGEQU 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> ; 20/11/2023
   204                                  <1> ; (playwav2.com, ac97.inc, Erdogan Tan, 11/11/2023)
   205                                  <1> ; 03/11/2023 - Erdogan Tan (Ref: MenuetOS AC97 WAV Player source code, 2004)
   206                                  <1> SIS_VID             equ     1039h
   207                                  <1> NVIDIA_VID  equ     10DEh    ; Ref: MPXPLAY/SBEMU/KOLIBRIOS AC97 source c.
   208                                  <1> AMD_VID             equ     1022h
   209                                  <1> ;
   210                                  <1> ICH_DID             equ     2415h   ; ICH (82801AA) device ID
   211                                  <1> ; 20/11/2023
   212                                  <1> ; (playwav2.com, ac97.inc, Erdogan Tan, 11/11/2023)
   213                                  <1> ; 17/02/2017 (Erdogan Tan, ref: ALSA Device IDs, ALSA project)
   214                                  <1> ICH0_DID        equ     2425h     ; ICH0
   215                                  <1> ICH2_DID        equ     2445h     ; ICH2
   216                                  <1> ICH3_DID    equ     2485h  ; ICH3
   217                                  <1> ICH4_DID        equ     24C5h     ; ICH4
   218                                  <1> ICH5_DID    equ     24D5h  ; ICH5
   219                                  <1> ICH6_DID    equ     266Eh  ; ICH6
   220                                  <1> ESB6300_DID equ     25A6h  ; 6300ESB
   221                                  <1> ESB631X_DID equ     2698h  ; 631XESB
   222                                  <1> ICH7_DID    equ     27DEh   ; ICH7
   223                                  <1> ; 03/11/2023 - Erdogan Tan (Ref: MenuetOS AC97 WAV Player source code, 2004)
```

```
                        <1> MX82440_DID equ     7195h
                        <1> SI7012_DID  equ     7012h
                        <1> NFORCE_DID  equ     01B1h
                        <1> NFORCE2_DID equ     006Ah
                        <1> AMD8111_DID equ     746Dh
                        <1> AMD768_DID  equ     7445h
                        <1> ; 03/11/2023 - Erdogan Tan - Ref: MPXPLAY/SBEMU/KOLIBRIOS AC97 source code
                        <1> CK804_DID   equ     0059h   ; NFORCE4
                        <1> MCP04_DID   equ     003Ah
                        <1> CK8_DID             equ     008Ah
                        <1> NFORCE3_DID equ     00DAh
                        <1> CK8S_DID    equ     00EAh
                        <1>
                        <1> NAMBAR_REG      equ       10h     ; native audio mixer Base Address Register
                        <1> NABMBAR_REG     equ       14h     ; native audio bus mastering Base Addr Reg
                        <1>
                        <1> PI_CR_REG       equ     0Bh     ; PCM in Control Register
                        <1> PO_CR_REG   equ     1Bh     ; PCM out Control Register
                        <1> MC_CR_REG   equ     2Bh     ; MIC in Control Register
                        <1>
                        <1> PI_SR_REG   equ     6       ; PCM in Status register
                        <1> PO_SR_REG   equ     16h     ; PCM out Status register
                        <1> MC_SR_REG   equ     26h     ; MIC in Status register
                        <1>
                        <1> IOCE                equ     BIT4    ; interrupt on complete enable.
                        <1> FEIFE               equ     BIT3    ; set if you want an interrupt to fire
                        <1> LVBIE               equ     BIT2    ; last valid buffer interrupt enable.
                        <1> RR          equ     BIT1    ; reset registers. Nukes all regs
                        <1>                             ; except bits 4:2 of this register.
                        <1>                             ; Only set this bit if BIT 0 is 0
                        <1> RPBM        equ     BIT0    ; Run/Pause
                        <1>                           ; set this bit to start the codec!
                        <1>
                        <1> PI_BDBAR_REG        equ     0       ; PCM in buffer descriptor BAR
                        <1> PO_BDBAR_REG        equ     10h     ; PCM out buffer descriptor BAR
                        <1> MC_BDBAR_REG        equ     20h     ; MIC in buffer descriptor BAR
                        <1>
                        <1> PI_CIV_REG  equ     4       ; PCM in current Index value (RO)
                        <1> PO_CIV_REG  equ     14h     ; PCM out current Index value (RO)
                        <1> MC_CIV_REG  equ     24h     ; MIC in current Index value (RO)
                        <1>
                        <1> PI_LVI_REG  equ     5       ; PCM in Last Valid Index
                        <1> PO_LVI_REG  equ     15h     ; PCM out Last Valid Index
                        <1> MC_LVI_REG  equ     25h     ; MIC in Last Valid Index
                        <1>
                        <1> IOC         equ     BIT31  ; Fire an interrupt whenever this
                        <1>                            ; buffer is complete.
                        <1> BUP         equ     BIT30  ; Buffer Underrun Policy.
                        <1>
                        <1> GLOB_CNT_REG        equ     2Ch     ; Global Control Register
                        <1> GLOB_STS_REG        equ     30h     ; Global Status register (RO)
                        <1>
                        <1> CTRL_ST_CREADY      equ   BIT8+BIT9+BIT28 ; Primary Codec Ready
                        <1>
                        <1> CODEC_REG_POWERDOWN   equ 26h
                        <1> CODEC_REG_ST          equ 26h
                        <1>
                        <1> ; 22/06/2017
                        <1> PO_PICB_REG equ 18h ; PCM Out Position In Current Buffer Register
                        <1>
                        <1> ; 19/11/2023
                        <1> AC97_EA_VRA equ BIT0
                        <1> ; 24/11/2023
                        <1> BCIS        equ BIT3  ; Buffer Completion Interrupt Status
                        <1>
                        <1> ;============================================================================
                        <1> ;                CODE
                        <1> ;============================================================================
                        <1>
                        <1> ; CODE for INTEL ICH AC'97 AUDIO CONTROLLER
                        <1>
                        <1> DetectICH:
                        <1>     ; 22/11/2023
                        <1>     ; 19/11/2023
                        <1>     ; 01/11/2023 - TRDOS 386 Kernel v2.0.7
                        <1>     ;; 10/06/2017
                        <1>     ;; 05/06/2017
                        <1>     ;; 29/05/2017
                        <1>     ;; 28/05/2017
                        <1>     ;mov    eax, (ICH_DID << 16) + INTEL_VID
                        <1>         ;call       pciFindDevice
                        <1>         ;jnc        short d_ac97_1
                        <1>     ;
                        <1>     ;; 01/11/2023
                        <1>     ;mov    eax, (NFORCE_DID << 16) + NFORCE_VID
                        <1>         ;call       pciFindDevice
                        <1>         ;jnc        short d_ac97_1
                        <1>
                        <1>     ; 19/11/2023
00013958 BE[7B470100]   <1>     mov     esi, valid_ids ; address of Valid ICH (AC97) Device IDs
0001395D B915000000     <1>     mov     ecx, valid_id_count
                        <1> pfd_1:
00013962 AD             <1>     lodsd
00013963 E857000000     <1>     call    pciFindDevice
00013968 730F           <1>     jnc     short d_ac97_1
0001396A E2F6           <1>     loop    pfd_1
                        <1>     ;stc
                        <1>
                        <1> d_ac97_0:
                        <1> ; couldn't find the audio device!
0001396C C3             <1>     retn
                        <1>
                        <1> ; CODE for VIA VT8233 AUDIO CONTROLLER
                        <1>
                        <1> DetectVT8233:
                        <1>     ; 22/11/2023
                        <1>     ; 02/11/2023
                        <1>     ; 01/11/2023 - TRDOS 386 Kernel v2.0.7
                        <1>     ; 06/08/2022 - TRDOS 386 Kernel v2.0.5
                        <1>     ; 10/06/2017
                        <1>     ; 05/06/2017
                        <1>     ; 29/05/2017
                        <1>     ; 03/04/2017
0001396D B806115930     <1>     mov     eax, (VT8233_DID << 16) + VIA_VID
00013972 E848000000     <1>         call    pciFindDevice
                        <1> ;       jnc     short d_vt8233_0
                        <1> ; couldn't find the audio device!
                        <1> ;   retn
00013977 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>
00013979 A3[B8880100]           <1>     mov     [audio_dev_id], eax
0001397E 8915[BC880100]         <1>     mov     [audio_vendor], edx
                                <1>
                                <1>     ; 22/11/2023
                                <1>     ;; init controller
                                <1>     ;mov    al, PCI_CMD_REG ; command register (04h)
                                <1>     ;call   pciRegRead32
                                <1>     ;
                                <1>     ;; eax = BUS/DEV/FN/REG
                                <1>     ;; edx = STATUS/COMMAND
                                <1>     ;;      SSSSSSSSSSSSSSSSCCCCCCCCCCCCCCCC
                                <1>     ;mov    [audio_stats_cmd], edx
                                <1>
00013984 B010                   <1>     mov     al, PCI_IO_BASE ; IO base address register (10h)
                                <1>     ;mov    al, NAMBAR_REG ; Native Audio Mixer BAR (10h)
00013986 E8BF000000             <1>     call    pciRegRead32
                                <1>
                                <1>     ;cmp    word [audio_vendor], INTEL_VID ; 8086h ; AC'97 ?
                                <1>     ;;jne   short d_vt8233_1
                                <1>     ;; 01/11/2023
                                <1>     ;je     short d_ac97_3
                                <1>     ;cmp    word [audio_vendor], NFORCE_VID ; 10DEh ; AC'97
                                <1>     ;jne    short d_vt8233_1
                                <1>     ; 02/11/2023
0001398B 66813D[BC880100]06-    <1>     cmp     word [audio_vendor], VIA_VID  ; 1106h ; VT8233-VT8237R
00013993 11                     <1>
00013994 7411                   <1>     je      short d_vt8233_1
                                <1> d_ac97_3:
                                <1>     ;and    dx, 0FFFEh ; Audio Codec IO_ADDR_MASK
                                <1>     ; 06/08/2022
00013996 80E2FE                 <1>     and     dl, 0FEh
00013999 668915[B4880100]       <1>     mov     [NAMBAR], dx
                                <1>
000139A0 B014                   <1>     mov     al, NABMBAR_REG ; Native Audio Bus Mastering BAR (14h)
000139A2 E8A3000000             <1>     call    pciRegRead32
                                <1>
                                <1>     ;;and   dx, 0FFC0h ; Audio Controller IO_ADDR_MASK
                                <1>     ;; 06/08/2022
                                <1>     ;and    dl, 0C0h
                                <1>     ;mov    [NABMBAR], dx
                                <1>        ;;mov      [audio_io_base], dx
                                <1>     ;
                                <1>     ;jmp    short d_ac97_2
                                <1>     ; 02/11/2023
                                <1>     ; NABMBAR = audio_io_base
                                <1>
                                <1> d_vt8233_1:
                                <1>     ;and    dx, 0FFC0h ; Audio Controller IO_ADDR_MASK
                                <1>     ; 06/08/2022
000139A7 80E2C0                 <1>     and     dl, 0C0h
000139AA 668915[B6880100]       <1>        mov     [audio_io_base], dx
                                <1>
                                <1> d_ac97_2:
                                <1>     ; 10/06/2017
000139B1 B03C                   <1>     mov     al, AC97_INT_LINE ; Interrupt Line Register (3Ch)
                                <1>     ;call   pciRegRead32
000139B3 E87F000000             <1>     call    pciRegRead8
                                <1>
                                <1>     ;;and   edx, 0FFh
                                <1>     ; 06/08/2022
                                <1>     ;and    dx, 0FFh
                                <1>
000139B8 8815[B2880100]         <1>     mov     [audio_intr], dl
                                <1>
000139BE 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>     ; 19/11/2023
                        <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, edi ; 19/11/2023
                        <1>
                        <1>     ;push   ecx
                        <1>     ;push   eax ; * ; 19/11/2023
                        <1>     ;push   esi
                        <1>     ;push   edi
                        <1>     ;
                        <1>     ;mov    esi, eax              ; save off vend+device ID
                        <1>         ; 19/11/2023
000139BF 89C3           <1>     mov     ebx, eax
                        <1>     ;mov    edi, (80000000h - 100h)       ; start with bus 0, dev 0 func 0
000139C1 BF00000080     <1>     mov     edi, 80000000h
                        <1> nextPCIdevice:
                        <1>     ;add    edi, 100h
                        <1>     ;cmp    edi, 80FFF800h        ; scanned all devices?
                        <1>     ;stc
                        <1>     ;je     short PCIScanExit     ; not found
                        <1>
000139C6 89F8           <1>     mov     eax, edi              ; read PCI registers
000139C8 E87D000000     <1>     call    pciRegRead32
                        <1>     ; 19/11/2023
000139CD 39DA           <1>     cmp     edx, ebx
                        <1>     ;cmp    edx, esi              ; found device?
                        <1>     ;jne    short nextPCIdevice
                        <1>     ;;clc
000139CF 7412           <1>     je      short PCIScanExit     ; found
                        <1>     ; 19/11/2023
000139D1 81FF00F8FF80   <1>     cmp     edi, 80FFF800h
000139D7 7308           <1>     jnb     short pfd_nf          ; not found
000139D9 81C700010000   <1>     add     edi, 100h
000139DF EBE5           <1>     jmp     short nextPCIdevice
                        <1> pfd_nf:
000139E1 F9             <1>     stc
000139E2 C3             <1>     retn
                        <1> PCIScanExit:
                        <1>     ;pushf
000139E3 B8FFFFFF7F     <1>     mov     eax, NOT_BIT31        ; 19/03/2017
000139E8 21F8           <1>     and     eax, edi       ; return only bus/dev/fn #
                        <1>     ;popf
                        <1>     ;
                        <1>     ;pop    edi
                        <1>     ;pop    esi
                        <1>     ;pop    edx ; * ; 19/11/2023
                        <1>     ;pop    ecx
                        <1>     ;
000139EA C3             <1>     retn
                        <1>
                        <1> ; 26/11/2023
                        <1> %if 1
                        <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>
000139EB 53             <1>     push    ebx
000139EC 51             <1>     push    ecx
000139ED 89C3           <1>         mov     ebx, eax          ; save eax, dh
000139EF 88F1           <1>         mov     cl, dh
                        <1>
000139F1 25FFFFFF3F     <1>         and     eax, NOT_PCI32_PCI16     ; clear out data size request
000139F6 0D00000080     <1>         or      eax, BIT31        ; make a PCI access request
000139FB 24FC           <1>         and     al, ~3 ; NOT 3            ; force index to be dword
                        <1>
000139FD 66BAF80C       <1>         mov     dx, PCI_INDEX_PORT
00013A01 EF             <1>         out dx, eax               ; write PCI selector
                        <1>
00013A02 66BAFC0C       <1>         mov     dx, PCI_DATA_PORT
00013A06 88D8           <1>         mov     al, bl
00013A08 2403           <1>         and     al, 3                     ; figure out which port to
00013A0A 00C2           <1>         add     dl, al                    ; read to
                        <1>
00013A0C F7C3000000C0   <1>     test    ebx, PCI32+PCI16
00013A12 7507           <1>         jnz     short _pregr0
00013A14 EC             <1>     in      al, dx                ; return 8 bits of data
00013A15 88C2           <1>         mov dl, al
00013A17 88CE           <1>     mov     dh, cl                ; restore dh for 8 bit read
00013A19 EB12           <1>     jmp     short _pregr2
                        <1> _pregr0:
00013A1B F7C300000080   <1>     test    ebx, PCI32
00013A21 7507           <1>         jnz short _pregr1
00013A23 66ED           <1>     in      ax, dx
00013A25 6689C2         <1>         mov     dx, ax                    ; return 16 bits of data
00013A28 EB03           <1>     jmp     short _pregr2
```

```
                                <1> _pregr1:
00013A2A ED                     <1>     in      eax, dx                 ; return 32 bits of data
00013A2B 89C2                   <1>     mov     edx, eax
                                <1> _pregr2:
00013A2D 89D8                   <1>     mov     eax, ebx                ; restore eax
00013A2F 25FFFFFF3F             <1>         and     eax, NOT_PCI32_PCI16    ; clear out data size request
00013A34 59                     <1>     pop     ecx
00013A35 5B                     <1>     pop     ebx
00013A36 C3                     <1>     retn
                                <1>
                                <1> pciRegRead8:
00013A37 25FFFFFF3F             <1>         and     eax, NOT_PCI32_PCI16    ; set up 8 bit read size
00013A3C EBAD                   <1>         jmp     short pciRegRead  ; call generic PCI access
                                <1>
                                <1> pciRegRead16:
00013A3E 25FFFFFF3F             <1>         and     eax, NOT_PCI32_PCI16    ; set up 16 bit read size
00013A43 0D00000040             <1>         or      eax, PCI16        ; call generic PCI access
00013A48 EBA1                   <1>         jmp     short pciRegRead
                                <1>
                                <1> pciRegRead32:
00013A4A 25FFFFFF3F             <1>         and     eax, NOT_PCI32_PCI16    ; set up 32 bit read size
00013A4F 0D00000080             <1>         or      eax, PCI32        ; call generic PCI access
00013A54 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>
00013A56 53                     <1>     push    ebx
00013A57 51                     <1>     push    ecx
00013A58 89C3                   <1>         mov     ebx, eax          ; save eax, edx
00013A5A 89D1                   <1>         mov     ecx, edx
00013A5C 25FFFFFF3F             <1>     and     eax, NOT_PCI32_PCI16  ; clear out data size request
00013A61 0D00000080             <1>         or      eax, BIT31        ; make a PCI access request
00013A66 24FC                   <1>         and     al, ~3 ; NOT 3            ; force index to be dword
                                <1>
00013A68 66BAF80C               <1>         mov     dx, PCI_INDEX_PORT
00013A6C EF                     <1>         out dx, eax               ; write PCI selector
                                <1>
00013A6D 66BAFC0C               <1>         mov     dx, PCI_DATA_PORT
00013A71 88D8                   <1>         mov     al, bl
00013A73 2403                   <1>         and     al, 3                     ; figure out which port to
00013A75 00C2                   <1>         add     dl, al                    ; write to
                                <1>
00013A77 F7C3000000C0           <1>     test    ebx, PCI32+PCI16
00013A7D 7505                   <1>         jnz     short _pregw0
00013A7F 88C8                   <1>     mov     al, cl                ; put data into al
00013A81 EE                     <1>     out     dx, al
00013A82 EB12                   <1>     jmp     short _pregw2
                                <1> _pregw0:
00013A84 F7C300000080           <1>     test    ebx, PCI32
00013A8A 7507                   <1>         jnz     short _pregw1
00013A8C 6689C8                 <1>     mov     ax, cx                ; put data into ax
00013A8F 66EF                   <1>     out     dx, ax
00013A91 EB03                   <1>     jmp     short _pregw2
                                <1> _pregw1:
00013A93 89C8                   <1>     mov     eax, ecx              ; put data into eax
00013A95 EF                     <1>     out     dx, eax
                                <1> _pregw2:
00013A96 89D8                   <1>         mov     eax, ebx          ; restore eax
00013A98 25FFFFFF3F             <1>         and     eax, NOT_PCI32_PCI16    ; clear out data size request
00013A9D 89CA                   <1>         mov     edx, ecx          ; restore dx
00013A9F 59                     <1>     pop     ecx
00013AA0 5B                     <1>     pop     ebx
00013AA1 C3                     <1>     retn
                                <1>
                                <1> pciRegWrite8:
00013AA2 25FFFFFF3F             <1>         and     eax, NOT_PCI32_PCI16    ; set up 8 bit write size
00013AA7 EBAD                   <1>         jmp short pciRegWrite     ; call generic PCI access
                                <1>
                                <1> pciRegWrite16:
00013AA9 25FFFFFF3F             <1>         and     eax, NOT_PCI32_PCI16    ; set up 16 bit write size
00013AAE 0D00000040             <1>         or      eax, PCI16        ; call generic PCI access
00013AB3 EBA1                   <1>         jmp short pciRegWrite
                                <1>
                                <1> pciRegWrite32:
00013AB5 25FFFFFF3F             <1>         and     eax, NOT_PCI32_PCI16    ; set up 32 bit write size
00013ABA 0D00000080             <1>         or      eax, PCI32        ; call generic PCI access
00013ABF EB95                   <1>         jmp pciRegWrite
                                <1>
                                <1> %endif
                                <1>
                                <1> ; 26/11/2023 - temporary
                                <1> %if 0
                                <1>
                                <1> ; PLAYWAV3.COM, ac97_vra.asm, 19/11/2023, Erdogan Tan
                                <1>
                                <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> ; Note: this routine is meant to be called via pciRegRead8, pciRegread16,
                                <1> ;   or pciRegRead32, listed below.
                                <1> ;
                                <1> ; Note2: don't attempt to read 32bits of data from a non dword aligned reg
                                <1> ;    number. Likewise, don't do 16bit reads from non word aligned reg #
                                <1> ;
                                <1> pciRegRead:
                                <1>     push    ebx
                                <1>     push    ecx
                                <1>         mov     ebx, eax                      ; save eax, dh
                                <1>         mov     cl, dh
                                <1>         and     eax, (~PCI32)+PCI16           ; clear out data size request
                                <1>         or      eax, BIT31                    ; make a PCI access request
```

```
                        <1>         and     al, ~3 ; NOT 3                  ; force index to be dword
                        <1>
                        <1>         mov     dx, PCI_INDEX_PORT
                        <1>         out     dx, eax                         ; write PCI selector
                        <1>
                        <1>         mov     dx, PCI_DATA_PORT
                        <1>         mov     al, bl
                        <1>         and     al, 3                           ; figure out which port to
                        <1>         add     dl, al                          ; read to
                        <1>
                        <1>     in      eax, dx                         ; do 32bit read
                        <1>         test    ebx, PCI32
                        <1>         jz      short _pregr1
                        <1>
                        <1>         mov     edx, eax                        ; return 32bits of data
                        <1> _pregr1:
                        <1>     mov     dx, ax                          ; return 16bits of data
                        <1>         test    ebx, PCI32+PCI16
                        <1>         jnz     short _pregr2
                        <1>         mov     dh, cl                          ; restore dh for 8 bit read
                        <1> _pregr2:
                        <1>         mov     eax, ebx                        ; restore eax
                        <1>         and     eax, (~PCI32)+PCI16             ; clear out data size request
                        <1>     pop     ecx
                        <1>     pop     ebx
                        <1>     retn
                        <1>
                        <1> pciRegRead8:
                        <1>         and     eax, (~PCI16)+PCI32             ; set up 8 bit read size
                        <1>         jmp     short pciRegRead        ; call generic PCI access
                        <1>
                        <1> pciRegRead16:
                        <1>         and     eax, (~PCI16)+PCI32             ; set up 16 bit read size
                        <1>         or      eax, PCI16              ; call generic PCI access
                        <1>         jmp     short pciRegRead
                        <1>
                        <1> pciRegRead32:
                        <1>         and     eax, (~PCI16)+PCI32             ; set up 32 bit read size
                        <1>         or      eax, PCI32              ; call generic PCI access
                        <1>         jmp     short pciRegRead
                        <1>
                        <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> ;
                        <1> ; note: this routine is meant to be called via pciRegWrite8, pciRegWrite16,
                        <1> ;   or pciRegWrite32 as detailed below.
                        <1> ;
                        <1> ; Note2: don't attempt to write 32bits of data from a non dword aligned reg
                        <1> ;    number. Likewise, don't do 16bit writes from non word aligned reg #
                        <1> ;
                        <1> pciRegWrite:
                        <1>     push    ebx
                        <1>     push    ecx
                        <1>         mov     ebx, eax                        ; save eax, dx
                        <1>         mov     cx, dx
                        <1>         or      eax, BIT31                      ; make a PCI access request
                        <1>         and     eax, ~PCI16 ; NOT PCI16         ; clear out data size request
                        <1>         and     al, ~3 ; NOT 3                  ; force index to be dword
                        <1>
                        <1>         mov     dx, PCI_INDEX_PORT
                        <1>         out     dx, eax                         ; write PCI selector
                        <1>
                        <1>         mov     dx, PCI_DATA_PORT
                        <1>         mov     al, bl
                        <1>         and     al, 3                           ; figure out which port to
                        <1>         add     dl, al                          ; write to
                        <1>
                        <1>         mov     eax, edx                        ; put data into eax
                        <1>         mov     ax, cx
                        <1>
                        <1>         out     dx, al
                        <1>         test    ebx, PCI16+PCI32                ; only 8bit access? bail
                        <1>         jz      short _pregw1
                        <1>
                        <1>         out     dx, ax                          ; write 16 bit value
                        <1>         test    ebx, PCI16                      ; 16bit requested?  bail
                        <1>         jnz     short _pregw1
                        <1>
                        <1>         out     dx, eax                         ; write full 32bit
                        <1> _pregw1:
                        <1>         mov     eax, ebx                        ; restore eax
                        <1>         and     eax, (~PCI32)+PCI16             ; clear out data size request
                        <1>         mov     dx, cx                          ; restore dx
                        <1>     pop     ecx
                        <1>     pop     ebx
                        <1>     ret
                        <1>
                        <1> pciRegWrite8:
                        <1>         and     eax, (~PCI16)+PCI32             ; set up 8 bit write size
                        <1>         jmp     short pciRegWrite       ; call generic PCI access
                        <1>
                        <1> pciRegWrite16:
                        <1>         and     eax, (~PCI16)+PCI32             ; set up 16 bit write size
                        <1>         or      eax, PCI16              ; call generic PCI access
                        <1>         jmp     short pciRegWrite
                        <1>
                        <1> pciRegWrite32:
                        <1>         and     eax, (~PCI16)+PCI32             ; set up 32 bit write size
                        <1>         or      eax, PCI32              ; call generic PCI access
                        <1>         jmp     short pciRegWrite
                        <1>
                        <1> ;=============================================================
                        <1>
                        <1> %endif
                        <1>
                        <1> init_codec:
                        <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
                        <1>     ; 05/06/2017
                        <1>     ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
                        <1>     ;
00013AC1 A1[B8880100]   <1>     mov     eax, [audio_dev_id]
00013AC6 B041           <1>     mov     al, VIA_ACLINK_CTRL
00013AC8 E86AFFFFFF     <1>     call    pciRegRead8
                        <1>     ; ?
00013ACD B040           <1>     mov     al, VIA_ACLINK_STAT
00013ACF E863FFFFFF     <1>     call    pciRegRead8
00013AD4 F6C201         <1>     test    dl, VIA_ACLINK_C00_READY
```

```
00013AD7 7508              <1>         jnz     short _codec_ready_1
00013AD9 E80D000000        <1>     call    reset_codec
00013ADE 7305              <1>     jnc     short _codec_ready_2 ; eax = 1
00013AE0 C3                <1>     retn
                           <1> _codec_ready_1:
                           <1>     ;mov    eax, 1
                           <1>     ; 06/08/2022
00013AE1 29C0              <1>     sub     eax, eax
00013AE3 FEC0              <1>     inc     al
                           <1>     ; eax = 1
                           <1> _codec_ready_2:
00013AE5 E880000000        <1>     call    codec_io_w16
                           <1> detect_codec:
00013AEA 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)
00013AEB A1[B8880100]      <1>     mov     eax, [audio_dev_id]
00013AF0 B041              <1>     mov     al, VIA_ACLINK_CTRL
00013AF2 B2E0              <1>             mov     dl, VIA_ACLINK_CTRL_ENABLE + VIA_ACLINK_CTRL_RESET + VIA_ACLINK_CTRL_SYNC
00013AF4 E8A9FFFFFF        <1>     call    pciRegWrite8
                           <1>
00013AF9 E843000000        <1>     call    delay_100ms    ; wait 100 ms
                           <1> _rc_cold:
00013AFE E811000000        <1>         call    cold_reset
00013B03 7301              <1>         jnc     short _reset_codec_ok
                           <1>
                           <1>     ; 16/04/2017
                           <1>         ;xor       eax, eax       ; timeout error
                           <1>             ;stc
00013B05 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>
00013B06 29C0              <1>     sub     eax, eax
                           <1>     ; 02/11/2023
                           <1>     ;mov    edx, CODEC_RESET_REG ; 00h ; Reset register
00013B08 31D2              <1>     xor     edx, edx ; 00h ; Reset register
00013B0A E8BE000000        <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>
00013B0F 31C0              <1>         xor     eax, eax
                           <1>         ;mov       al, VIA_ACLINK_C00_READY ; 1
00013B11 FEC0              <1>         inc al
00013B13 C3                <1>     retn
                           <1>
                           <1> cold_reset:
                           <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
                           <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
00013B14 30D2              <1>     xor     dl, dl ; 0
00013B16 E887FFFFFF        <1>     call    pciRegWrite8
                           <1>
00013B1B E821000000        <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
00013B20 B2CC              <1>     mov     dl, VIA_ACLINK_CTRL_INIT
00013B22 E87BFFFFFF        <1>     call    pciRegWrite8
                           <1>
                           <1>     ;mov    ecx, 16 ; total 2s
                           <1>     ; 06/08/2022
                           <1>     ;sub    ecx, ecx
                           <1>     ; 22/11/2023
                           <1>     ; ecx = 0 (from 'delay_100ms')
00013B27 B110              <1>     mov     cl, 16
                           <1> _crst_wait:
                           <1>     ;mov    eax, [audio_dev_id]
00013B29 B040              <1>     mov     al, VIA_ACLINK_STAT
00013B2B E807FFFFFF        <1>     call    pciRegRead8
                           <1>
00013B30 F6C201            <1>         test    dl, VIA_ACLINK_C00_READY
00013B33 750B              <1>         jnz     short _crst_ok
                           <1>
00013B35 51                <1>     push    ecx
00013B36 E806000000        <1>     call    delay_100ms
00013B3B 59                <1>     pop     ecx
                           <1>
00013B3C 49                <1>         dec     ecx
00013B3D 75EA              <1>         jnz     short _crst_wait
                           <1>
                           <1> _crst_fail:
00013B3F F9                <1>         stc
                           <1> _crst_ok:
00013B40 C3                <1>     retn
                           <1>
                           <1> delay_100ms:
                           <1>     ; 29/05/2017
                           <1>     ; 24/03/2017 ('codec.asm')
                           <1>     ; wait 100 ms
00013B41 B990010000        <1>     mov     ecx, 400  ; 400*0.25ms
                           <1> _delay_x_ms:
00013B46 E803000000        <1>     call    delay1_4ms
00013B4B E2F9              <1>         loop       _delay_x_ms
00013B4D 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>     ; 29/05/2017
                        <1>     ; 23/04/2017
                        <1>     ; 05/03/2017 (TRDOS 386)
                        <1>     ; ('UTILS.ASM')
                        <1> delay1_4ms:
00013B4E 50             <1>         push    eax
00013B4F 51             <1>         push    ecx
00013B50 B110           <1>         mov cl, 16          ; close enough.
                        <1>
00013B52 E461           <1>     in      al, PORTB ; 61h
                        <1>
00013B54 2410           <1>     and     al, REFRESH_STATUS ; 10h
00013B56 88C5           <1>     mov     ch, al          ; Start toggle state
                        <1> _d4ms1:
00013B58 E461           <1>     in      al, PORTB       ; Read system control port
                        <1>
00013B5A 2410           <1>     and     al, REFRESH_STATUS ; Refresh toggles 15.085 microseconds
00013B5C 38C5           <1>     cmp     ch, al
00013B5E 74F8           <1>     je      short _d4ms1    ; Wait for state change
                        <1>
00013B60 88C5           <1>     mov     ch, al          ; Update with new state
00013B62 FEC9           <1>     dec     cl
00013B64 75F2           <1>     jnz     short _d4ms1
                        <1>
00013B66 F8             <1>     clc     ; 29/05/2017
                        <1>
00013B67 59             <1>         pop     ecx
00013B68 58             <1>         pop     eax
00013B69 C3             <1>         retn
                        <1>
                        <1> ; 10/04/2017 (TRDOS 386)
                        <1> ; 12/11/2016
                        <1>
                        <1> codec_io_w16: ;w32
                        <1>     ; ('codec.asm')
00013B6A 668B15[B6880100]   <1>         mov dx, [audio_io_base]
00013B71 6681C28000         <1>         add     dx, VIA_REG_AC97
00013B76 EF             <1>     out     dx, eax
00013B77 C3             <1>         retn
                        <1>
                        <1> codec_io_r16: ;r32
                        <1>     ; ('codec.asm')
00013B78 668B15[B6880100]   <1>         mov     dx, [audio_io_base]
00013B7F 6681C28000         <1>         add     dx, VIA_REG_AC97
00013B84 ED             <1>         in  eax, dx
00013B85 C3             <1>         retn
                        <1>
                        <1> ctrl_io_w8:
                        <1>     ; ('codec.asm')
00013B86 660315[B6880100]   <1>         add     dx, [audio_io_base]
00013B8D EE             <1>         out dx, al
00013B8E C3             <1>         retn
                        <1>
                        <1> ctrl_io_r8:
                        <1>     ; ('codec.asm')
00013B8F 660315[B6880100]   <1>         add     dx, [audio_io_base]
00013B96 EC             <1>         in  al, dx
00013B97 C3             <1>         retn
                        <1>
                        <1> ctrl_io_w32:
                        <1>     ; ('codec.asm')
00013B98 660315[B6880100]   <1>         add     dx, [audio_io_base]
00013B9F EF             <1>         out dx, eax
00013BA0 C3             <1>         retn
                        <1>
                        <1> ctrl_io_r32:
                        <1>     ; ('codec.asm')
00013BA1 660315[B6880100]   <1>         add dx, [audio_io_base]
00013BA8 ED             <1>     in      eax, dx
                        <1> _cr_not_rdy:        ; 06/08/2022
00013BA9 C3             <1>         retn
                        <1>
                        <1> codec_read:
                        <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
                        <1>     ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
                        <1>         ; Use only primary codec.
                        <1>         ; eax = register
00013BAA C1E010         <1>         shl     eax, VIA_REG_AC97_CMD_SHIFT
00013BAD 0D00008002     <1>         or      eax, VIA_REG_AC97_PRIMARY_VALID + VIA_REG_AC97_READ
                        <1>
00013BB2 E8B3FFFFFF     <1>     call    codec_io_w16
                        <1>
                        <1>             ; codec_valid
00013BB7 E825000000     <1>     call    codec_check_ready
                        <1>     ;jnc    short _cr_ok
                        <1>     ;retn
                        <1>     ; 06/08/2022
00013BBC 72EB           <1>     jc      short _cr_not_rdy
                        <1>     ; ecx <= 20
                        <1> _cr_ok:
                        <1>     ; wait 25 ms
                        <1>     ;mov    ecx, 80 ; (100*0.25 ms)
                        <1>     ; 06/08/2022
                        <1>     ;xor    ecx, ecx
00013BBE B150           <1>     mov     cl, 80
                        <1>     ; ecx = 80
                        <1> _cr_wloop:
00013BC0 E889FFFFFF     <1>     call    delay1_4ms
00013BC5 E2F9           <1>     loop    _cr_wloop
                        <1>
00013BC7 E8ACFFFFFF     <1>         call    codec_io_r16
                        <1>     ; 06/08/2022
                        <1>     ;and    eax, 0FFFFh
00013BCC C3             <1>         retn
                        <1>
                        <1> codec_write:
                        <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
                        <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>
00013BCD C1E210         <1>     shl     edx, VIA_REG_AC97_CMD_SHIFT
                        <1>
                        <1>     ; 02/11/2023 (shl eax, 0)
                        <1>     ;shl    eax, VIA_REG_AC97_DATA_SHIFT ; shl eax, 0
00013BD0 09C2           <1>         or      edx, eax
```

```
                                <1>
00013BD2 B800000000             <1>         mov     eax, VIA_REG_AC97_CODEC_ID_PRIMARY
00013BD7 C1E01E                 <1>         shl     eax, VIA_REG_AC97_CODEC_ID_SHIFT
00013BDA 09D0                   <1>         or      eax, edx
                                <1>
00013BDC E889FFFFFF             <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>     ; 06/08/2022 - TRDOS 386 v2.0.5
                                <1>     ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
                                <1>
                                <1> _codec_check_ready:
                                <1>     ;mov    ecx, 20 ; total 2s
                                <1>     ; 06/08/2022
00013BE1 29C9                   <1>     sub     ecx, ecx
                                <1>     ;mov    cl, 20
                                <1>     ; 02/11/2023
00013BE3 B10A                   <1>     mov     cl, 10  ; wait 1s
                                <1> _ccr_wait:
00013BE5 51                     <1>     push    ecx
                                <1>
00013BE6 E88DFFFFFF             <1>         call    codec_io_r16
00013BEB A900000001             <1>         test    eax, VIA_REG_AC97_BUSY
00013BF0 740B                   <1>         jz      short _ccr_ok
                                <1>
00013BF2 E84AFFFFFF             <1>     call    delay_100ms
                                <1>
00013BF7 59                     <1>     pop     ecx
                                <1>
00013BF8 49                     <1>     dec      ecx
00013BF9 75EA                   <1>         jnz     short _ccr_wait
                                <1>
00013BFB F9                     <1>         stc
00013BFC C3                     <1>         retn
                                <1>
                                <1> _ccr_ok:
00013BFD 59                     <1>     pop     ecx
00013BFE 25FFFF0000             <1>     and      eax, 0FFFFh
00013C03 C3                     <1>         retn
                                <1>
                                <1> codec_config:
                                <1>     ; 02/11/2023 - TRDOS 386 Kernel v2.0.7
                                <1>     ; 06/08/2022 - TRDOS 386 Kernel v2.0.5
                                <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>
00013C04 B802020000             <1>     mov     eax, 0202h
00013C09 66A3[EC880100]         <1>     mov     [audio_master_volume], ax
                                <1>
                                <1>     ;mov    ax, 1F1Fh ; 31,31
                                <1>     ; 02/11/2023
00013C0F 66B80B0B               <1>     mov     ax, 0B0Bh
                                <1>     ;mov    edx, CODEC_MASTER_VOL_REG ; 02h ; Line Out
                                <1>     ; 06/08/2022
00013C13 29D2                   <1>     sub     edx, edx
00013C15 B202                   <1>     mov     dl, CODEC_MASTER_VOL_REG ; 02h ; Line Out
00013C17 E8B1FFFFFF             <1>     call    codec_write
                                <1>     ;jc     short cconfig_error
                                <1>
                                <1>     ;mov    eax, 0202h
00013C1C 66B80202               <1>     mov     ax, 0202h
                                <1>     ;mov    edx, CODEC_PCM_OUT_REG ; 18h ; Wave Output (Stereo)
                                <1>     ; 06/08/2022
00013C20 29D2                   <1>     sub     edx, edx
00013C22 B218                   <1>     mov     dl, CODEC_PCM_OUT_REG ; 18h ; Wave Output (Stereo)
00013C24 E8A4FFFFFF             <1>     call    codec_write
                                <1>     ;jc     short cconfig_error
                                <1>
                                <1>     ;mov    eax, 0202h
00013C29 66B80202               <1>     mov     ax, 0202h
                                <1>     ;mov    edx, CODEC_AUX_VOL ; 04h ; CODEC_HP_VOL_REG ; HeadPhone
                                <1>     ; 06/08/2022
00013C2D 29D2                   <1>     sub     edx, edx
00013C2F B204                   <1>     mov     dl, CODEC_AUX_VOL ; 04h ; CODEC_HP_VOL_REG ; HeadPhone
00013C31 E897FFFFFF             <1>     call    codec_write
                                <1>     ;jc     short cconfig_error
                                <1>
                                <1>     ;mov    eax, 08h
                                <1>         ;mov    ax, 08h
00013C36 66B80880               <1>     mov     ax, 8008h ; Mute
                                <1>     ;mov    edx, 0Ch  ; AC97_PHONE_VOL ; TAD Input (Mono)
                                <1>     ; 06/08/2022
00013C3A 29D2                   <1>     sub     edx, edx
00013C3C B20C                   <1>     mov     dl, 0Ch  ; AC97_PHONE_VOL ; TAD Input (Mono)
00013C3E E88AFFFFFF             <1>     call    codec_write
                                <1>     ;jc     short cconfig_error
                                <1>
                                <1>     ;mov    eax, 0808h
00013C43 66B80808               <1>     mov     ax, 0808h
                                <1>     ;mov    edx, CODEC_LINE_IN_VOL_REG ; 10h ; Line Input (Stereo)
                                <1>     ; 06/08/2022
00013C47 29D2                   <1>     sub     edx, edx
00013C49 B210                   <1>     mov     dl, CODEC_LINE_IN_VOL_REG ; 10h ; Line Input (Stereo)
00013C4B E87DFFFFFF             <1>     call    codec_write
                                <1>     ;jc     short cconfig_error
                                <1>
                                <1>     ;mov    eax, 0808h
00013C50 66B80808               <1>     mov     ax, 0808h
                                <1>     ;mov    edx, CODEC_CD_VOL_REG ; 12h ; CR Input (Stereo)
                                <1>     ; 06/08/2022
00013C54 29D2                   <1>     sub     edx, edx ; 02/11/2023
00013C56 B212                   <1>     mov     dl, CODEC_CD_VOL_REG ; 12h ; CR Input (Stereo)
00013C58 E870FFFFFF             <1>     call    codec_write
                                <1>     ;jc     short cconfig_error
                                <1>
                                <1>     ;mov    eax, 0808h
00013C5D 66B80808               <1>     mov     ax, 0808h
                                <1>         ;mov       edx, CODEC_AUX_VOL_REG ; 16h ; Aux Input (Stereo)
                                <1>     ; 06/08/2022
00013C61 29D2                   <1>     sub     edx, edx
00013C63 B216                   <1>     mov     dl, CODEC_AUX_VOL_REG ; 16h ; Aux Input (Stereo)
                                <1>     ;call   codec_write
```

```
1215                                <1>     ;;jc    short cconfig_error
1216 00013C65 E963FFFFFF           <1>     jmp     codec_write ; 10/06/2017
1217                                <1>
1218                                <1> ;   ; Extended Audio Status (2Ah)
1219                                <1> ;   mov     eax, CODEC_EXT_AUDIO_CTRL_REG ; 2Ah
1220                                <1> ;   call    codec_read
1221                                <1> ;   and     eax, 0FFFFh - 2             ; clear DRA (BIT1)
1222                                <1> ;   ;or     eax, 1                ; set VRA (BIT0)
1223                                <1> ;   or      eax, 5          ; VRA (BIT0) & S/PDIF (BIT2) ; 14/11/2016
1224                                <1> ;   mov     edx, CODEC_EXT_AUDIO_CTRL_REG
1225                                <1> ;   call    codec_write
1226                                <1> ;   ;jc     short cconfig_error
1227                                <1> ;
1228                                <1> ;set_sample_rate:
1229                                <1> ;   ;movzx  eax, word [audio_freq]
1230                                <1> ;   mov     ax, [audio_freq]
1231                                <1> ;   mov     edx, CODEC_PCM_FRONT_DACRATE_REG ; 2Ch ; PCM Front DAC Rate
1232                                <1> ;   ;call   codec_write
1233                                <1> ;   ;retn
1234                                <1> ;   jmp     codec_write
1235                                <1>
1236                                <1> ;cconfig_error:
1237                                <1> ;   retn
1238                                <1>
1239                                <1> vt8233_int_handler:
1240                                <1>     ; 05/06/2024
1241                                <1>     ; 04/06/2024 - TRDOS 386 v2.0.8
1242                                <1>     ; 27/07/2020
1243                                <1>     ; 22/07/2020
1244                                <1>     ; Interrupt Handler for VIA VT8237R Audio Controller
1245                                <1>     ; Note: called by 'dev_IRQ_service'
1246                                <1>     ; 14/10/2017
1247                                <1>     ; 09/10/2017, 10/10/2017, 12/10/2017
1248                                <1>     ; 13/06/2017
1249                                <1>     ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
1250                                <1>     ; 24/03/2017 - 'PLAYER.COM' ('player.asm')
1251                                <1>
1252                                <1>     ;push   eax ; * must be saved !
1253                                <1>     ;push   edx
1254                                <1>     ;push   ecx
1255                                <1>     ;push   ebx ; * must be saved !
1256                                <1>     ;push   esi
1257                                <1>     ;push   edi
1258                                <1>
1259                                <1>     ;cmp    byte [audio_busy], 1
1260                                <1>     ;jnb    short _ih0 ; 09/10/2017
1261                                <1>
1262                                <1>     ;mov    byte [audio_flag_eol], 0
1263                                <1>
1264 00013C6A 66BA0000             <1>         mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STATUS
1265 00013C6E E81CFFFFFF           <1>         call    ctrl_io_r8
1266                                <1>
1267 00013C73 A880                 <1>     test    al, VIA_REG_STAT_ACTIVE
1268 00013C75 7417                 <1>         jz      short _ih0 ; 09/10/2017
1269                                <1>
1270 00013C77 2407                 <1>         and     al, VIA_REG_STAT_EOL + VIA_REG_STAT_FLAG + VIA_REG_STAT_STOPPED
1271 00013C79 A2[EB880100]         <1>     mov     [audio_flag_eol], al
1272 00013C7E 740E                 <1>         jz  short _ih0 ; 09/10/2017
1273                                <1>
1274                                <1>     ; 09/10/2017
1275                                <1>     ;mov    byte [audio_busy], 1
1276                                <1>
1277 00013C80 803D[EA880100]01     <1>     cmp     byte [audio_play_cmd], 1
1278 00013C87 7315                 <1>     jnb     short _ih1 ; 10/10/2017
1279                                <1>
1280 00013C89 E848000000           <1>     call    channel_reset
1281                                <1> _ih0:
1282                                <1>     ; 09/10/2017
1283 00013C8E A0[EB880100]         <1>         mov     al, [audio_flag_eol]   ;; ack ;;
1284 00013C93 66BA0000             <1>         mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STATUS
1285 00013C97 E8EAFEFFFF           <1>         call    ctrl_io_w8
1286 00013C9C EB37                 <1>     jmp     short _ih4
1287                                <1> _ih1:
1288                                <1> vt8233_tuneLoop:
1289 00013C9E A0[EB880100]         <1>         mov     al, [audio_flag_eol]   ;; ack ;;
1290 00013CA3 66BA0000             <1>         mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STATUS
1291 00013CA7 E8DAFEFFFF           <1>         call    ctrl_io_w8
1292                                <1>
1293                                <1>     ; 22/07/2020
1294                                <1>     ;; 12/10/2017
1295                                <1>     ;mov    byte [audio_flag], 0 ; Reset
1296                                <1>
1297                                <1>     ; 10/10/2017
1298                                <1>     ; 09/10/2017
1299                                <1>     ;test   byte [audio_flag_eol], VIA_REG_STAT_FLAG
1300                                <1>     ;jz     short _ih2 ; EOL
1301                                <1>
1302                                <1>     ; 22/07/2020
1303                                <1>     ; 14/10/2017
1304                                <1>     ;test   byte [audio_flag_eol], VIA_REG_STAT_EOL
1305                                <1>     ;jnz    short _ih2 ; EOL
1306                                <1>     ;                   ; (Half Buffer 2 has been completed
1307                                <1>     ;                   ; and Half Buffer 1 will be played.)
1308                                <1>
1309                                <1>     ; FLAG
1310                                <1>     ; (Half Buffer 1 has been completed
1311                                <1>     ;  and Half Buffer 2 will be played.)
1312                                <1>
1313                                <1>     ; 14/10/2017
1314                                <1>     ;; (Continue to play.)
1315                                <1>     ;mov    al, VIA_REG_CTRL_INT
1316                                <1>             ;or     al, VIA_REG_CTRL_START
1317                                <1>             ;mov    dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
1318                                <1>         ;call       ctrl_io_w8
1319                                <1>     ; 12/10/2017
1320                                <1>     ;mov    byte [audio_flag], 1
1321                                <1>
1322                                <1>     ; 22/07/2020
1323                                <1>     ;inc    byte [audio_flag] ; = 1
1324                                <1> _ih2:
1325                                <1>     ; 10/10/2017
1326 00013CAC 8B3D[D0880100]       <1>     mov     edi, [audio_dma_buff]
1327                                <1>
1328                                <1>     ; 05/06/2024
1329                                <1>     ; 04/06/2024
1330                                <1>     ;;mov   ecx, [audio_dmabuff_size]
1331                                <1>     ;;shr   ecx, 1 ; dma buff size / 2 = half buffer size
1332                                <1>     ;mov    ecx, [audio_buff_size]
1333                                <1>     ;and    cl, ~1 ; word aligned
1334                                <1>     ; 05/06/2024
1335 00013CB2 8B0D[D8880100]       <1>     mov     ecx, [dma_hbuff_size] ; half buffer size
1336                                <1>
1337                                <1>     ; 22/07/2020
1338                                <1>     ; 12/10/2017
```

```
                                <1>     ;cmp    byte [audio_flag], 0
                                <1>     ;ja     short _ih3 ; Playing Half Buffer 2 (Current: FLAG)
                                <1>
                                <1>     ; 27/07/2020
                                <1>     ; 22/07/2020
00013CB8 F605[DC880100]01       <1>     test    byte [audio_flag], 1  ; Current flag value
00013CBF 7402                   <1>     jz      short _ih3 ; Half Buffer 1 must be filled
                                <1>
                                <1>     ; Half Buffer 2 must be filled
00013CC1 01CF                   <1>     add     edi, ecx
                                <1> _ih3:
                                <1>     ; Update half buffer 2 while playing half buffer 1
                                <1>     ; Update half buffer 1 while playing half buffer 2
                                <1>
00013CC3 8B35[C8880100]         <1>     mov     esi, [audio_p_buffer] ; phy addr of audio buff
                                <1>     ; 05/06/2024
                                <1>     ;shr    ecx, 2 ; half buff size / 4
                                <1>     ;rep    movsd
00013CC9 D1E9                   <1>     shr     ecx, 1  ; word aligned buffer
00013CCB F366A5                 <1>     rep     movsw
                                <1>
                                <1>     ; switch flag value ;
00013CCE 8035[DC880100]01       <1>     xor     byte [audio_flag], 1
                                <1>     ; 12/10/2017
                                <1>     ; [audio_flag] = 0 : Playing dma half buffer 2
                                <1>                        ; Next buffer (to update) is dma half buff 1
                                <1>     ;              = 1 : Playing dma half buffer 1
                                <1>                        ; Next buffer (to update) is dma half buff 2
                                <1> _ih4:
                                <1>     ; 28/05/2017
                                <1>     ;mov    byte [audio_busy], 0 ; 09/10/2017
                                <1>     ;
                                <1>     ;pop    edi
                                <1>     ;pop    esi
                                <1>     ;pop    ebx ; * must be restored !
                                <1>     ;pop    ecx
                                <1>     ;pop    edx
                                <1>     ;pop    eax ; * must be restored !
                                <1>
00013CD5 C3                     <1>     retn
                                <1>
                                <1> channel_reset:
                                <1>     ; 06/08/2022 - TRDOS 386 Kernel v2.0.5
                                <1>     ; 24/06/2017
                                <1>     ; 29/05/2017
                                <1>     ; 23/03/2017
                                <1>     ; 14/11/2016 - Erdogan Tan
                                <1>     ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
                                <1>        ;mov       edx, VIA_REG_OFFSET_CONTROL
                                <1>     ; 06/08/2022
00013CD6 29D2                   <1>     sub     edx, edx
00013CD8 B201                   <1>     mov     dl, VIA_REG_OFFSET_CONTROL
                                <1>     ;mov    eax, VIA_REG_CTRL_PAUSE + VIA_REG_CTRL_TERMINATE + VIA_REG_CTRL_RESET
00013CDA B848000000             <1>        mov eax, VIA_REG_CTRL_PAUSE + VIA_REG_CTRL_TERMINATE ; 24/06/2017
00013CDF E8A2FEFFFF             <1>     call    ctrl_io_w8
                                <1>
                                <1>        ;mov       edx, VIA_REG_OFFSET_CONTROL
                                <1>        ;call   ctrl_io_r8
                                <1>
                                <1>     ; wait for 50 ms
                                <1>     ;mov    ecx, 160 ; (200*0.25 ms) ; 29/05/2017
                                <1>     ; 06/08/2022
00013CE4 31C9                   <1>     xor     ecx, ecx
00013CE6 B1A0                   <1>     mov     cl, 160
                                <1> _ch_rst_wait:
00013CE8 E861FEFFFF             <1>     call    delay1_4ms
00013CED 49                     <1>     dec     ecx
00013CEE 75F8                   <1>     jnz     short _ch_rst_wait
                                <1>
                                <1>        ; disable interrupts
                                <1>     ;mov    edx, VIA_REG_OFFSET_CONTROL
                                <1>        ; 06/08/2022
00013CF0 29D2                   <1>     sub     edx, edx
00013CF2 B201                   <1>     mov     dl, VIA_REG_OFFSET_CONTROL
00013CF4 31C0                   <1>     xor     eax, eax
00013CF6 E88BFEFFFF             <1>        call    ctrl_io_w8
                                <1>
                                <1>        ; clear interrupts
                                <1>        ;mov       edx, VIA_REG_OFFSET_STATUS
                                <1>     ; 06/08/2022
00013CFB 29D2                   <1>     sub     edx, edx
                                <1>     ;mov    dl, VIA_REG_OFFSET_STATUS ; 0
                                <1>     ; edx = 0
                                <1>     ;mov    eax, 3
                                <1>        ; 06/08/2022
00013CFD 29C0                   <1>     sub     eax, eax
00013CFF B003                   <1>     mov     al, 3
                                <1>     ; eax = 3
00013D01 E880FEFFFF             <1>     call    ctrl_io_w8
                                <1>
                                <1>     ;mov    edx, VIA_REG_OFFSET_CURR_PTR
                                <1>     ;xor    eax, eax
                                <1>     ;call   ctrl_io_w32
                                <1>
00013D06 C3                     <1>        retn
                                <1>
                                <1> vt8233_stop: ; 22/04/2017
00013D07 C605[EA880100]00       <1>     mov     byte [audio_play_cmd], 0 ; stop !
                                <1> _tlp2:
                                <1>     ; 24/06/2017
                                <1>        ; finished with song, stop everything
                                <1>     ;mov    al, VIA_REG_CTRL_INT
                                <1>        ;or al, VIA_REG_CTRL_TERMINATE
                                <1>     ;mov    dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
                                <1>        ;call      ctrl_io_w8
                                <1>
                                <1>        ;call      channel_reset
                                <1>     ;retn
                                <1>
00013D0E EBC6                   <1>     jmp     short channel_reset
                                <1>
                                <1> set_vt8233_bdl: ; Set VT8237R Buffer Descriptor List
                                <1>     ; 04/06/2024 - TRDOS 386 v2.0.8
                                <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
                                <1>     ; 22/07/2020 - TRDOS 386 v2.0.2
                                <1>     ; 28/05/2017
                                <1>     ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
                                <1>     ; 24/03/2017 - 'PLAYER.COM' ('via_wav.asm' - 29/11/2016)
                                <1>
                                <1>     ; eax = dma buffer address = [audio_DMA_buff]
                                <1>     ; ecx = dma buffer buffer size = [audio_dmabuff_size]
                                <1>
                                <1>     ; 04/06/2024
                                <1>     ; ecx = DMA half buffer size (same as audio buffer size)
```

```
                                <1>
                                <1>     ;shr    ecx, 1 ; dma half buffer size
00013D10 89CE                   <1>     mov     esi, ecx
                                <1>
00013D12 BF[F0880100]           <1>         mov     edi, audio_bdl_buff      ; get BDL address
                                <1>     ;mov    ecx, 32 / 2             ; make 32 entries in BDL
                                <1>     ; 06/08/2022
00013D17 29C9                   <1>     sub     ecx, ecx
00013D19 B110                   <1>     mov     cl, 16
                                <1>
00013D1B EB05                   <1>     jmp     short s_vt8233_bdl1
                                <1>
                                <1> s_vt8233_bdl0:
                                <1>     ; set buffer descriptor 0 to start of data file in memory
                                <1>
00013D1D A1[D0880100]           <1>     mov     eax, [audio_dma_buff] ; Physical address of DMA buffer
                                <1>
                                <1> s_vt8233_bdl1:
00013D22 AB                     <1>     stosd                           ; store dmabuffer1 address
                                <1>
00013D23 89C2                   <1>     mov     edx, eax
                                <1>
                                <1> ; VIA VT8235.PDF: (Page 110) (Erdogan Tan, 29/11/2016)
                                <1>     ;
                                <1>     ;       Audio SGD Table Format
                                <1>     ;       ------------------------------
                                <1>     ;       63   62    61-56    55-32  31-0
                                <1>     ;       --   --   --------  -----  ----
                                <1>     ;       EOL FLAG -reserved- Base   Base
                                <1>     ;                           Count  Address
                                <1>     ;                            [23:0] [31:0]
                                <1>     ;       EOL: End Of Link.
                                <1>     ;            1 indicates this block is the last of the link.
                                <1>     ;            If the channel “Interrupt on EOL” bit is set, then
                                <1>     ;            an interrupt is generated at the end of the transfer.
                                <1>     ;
                                <1>     ;       FLAG: Block Flag. If set, transfer pauses at the end of this
                                <1>     ;             block. If the channel “Interrupt on FLAG” bit is set,
                                <1>     ;             then an interrupt is generated at the end of this block.
                                <1>
00013D25 89F0                   <1>     mov     eax, esi ; DMA half buffer size
                                <1>
                                <1>             ; 04/06/2024 - Erdogan Tan
                                <1>             ; NOTE: I have changed DMA half buffer size to
                                <1>             ; (word aligned) audio buffer size for smooth audio playing
                                <1>             ; (it was page border aligned before June 4, 2024)
                                <1>             ;
                                <1>             ; For example:
                                <1>             ;    For 65416 bytes audio buffer (22kHz, 16bit samples)
                                <1>             ;       (user's audio buff virtual addr is mapped to physical)
                                <1>             ;      ((kernel's and user's audio buff pages are same))
                                <1>             ;    1) Memory allocation (of it) is 65536 bytes
                                <1>             ;    2) DMA half buffer size is 65416 bytes
                                <1>             ;       (it would be 65536 bytes before this modification)
                                <1>             ;       ((additional 140 bytes would cause to a noise))
                                <1>             ;    3) Total DMA buffer size is 131072 bytes
                                <1>             ;       (the last 240 bytes will not be used for playing)
                                <1>
00013D27 01C2                   <1>     add     edx, eax
00013D29 0D00000040             <1>     or      eax, FLAG
                                <1>     ;or     eax, EOL
00013D2E AB                     <1>     stosd
                                <1>
                                <1> ; 2nd buffer:
                                <1>
00013D2F 89D0                   <1>         mov eax, edx ; Physical address of the 2nd half of DMA buffer
00013D31 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> ;
00013D32 89F0                   <1>     mov     eax, esi ; DMA half buffer size
                                <1>     ; 22/07/2020
                                <1>     ;or     eax, EOL
00013D34 0D00000040             <1>     or      eax, FLAG
00013D39 AB                     <1>     stosd
                                <1>
00013D3A E2E1                   <1>     loop    s_vt8233_bdl0
                                <1>
                                <1>     ; 22/07/2020
00013D3C 814FFC00000080         <1>     or      dword [edi-4], EOL
                                <1>
00013D43 C3                     <1>     retn
                                <1>
                                <1> vt8233_start_play:
                                <1>     ; 06/08/2022 - TRDOS 386 Kernel v2.0.5
                                <1>     ; 01/09/2020
                                <1>     ; 22/07/2020
                                <1>     ; start to play audio data via VT8233 audio controller
                                <1>     ; 13/06/2017
                                <1>     ; 10/06/2017
                                <1>     ; 24/04/2017
                                <1>     ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
                                <1>     ; 24/03/2017 - 'PLAYER.COM' ('via_wav.asm' - 29/11/2016)
                                <1>     ; write buffer descriptor list address
                                <1>
                                <1>     ; Extended Audio Status (2Ah)
00013D44 B82A000000             <1>     mov     eax, CODEC_EXT_AUDIO_CTRL_REG ; 2Ah
00013D49 E85CFEFFFF             <1>     call    codec_read
00013D4E 25FDFF0000             <1>     and     eax, 0FFFFh - 2             ; clear DRA (BIT1)
                                <1>     ;or     eax, 1                  ; set VRA (BIT0)
                                <1>     ;or     eax, 5          ; VRA (BIT0) & S/PDIF (BIT2) ; 14/11/2016
00013D53 0C05                   <1>     or      al, 5
                                <1>     ; 01/09/2020
                                <1>     ;or     eax, 3805h ; AD1980 (PRK, PRJ, PRI = 1 .. only front DAC)
                                <1>     ; 01/09/2020
                                <1>     ;mov    edx, CODEC_EXT_AUDIO_CTRL_REG
                                <1>     ;cmp    word [audio_freq], 0BB80h ; 48 kHz
                                <1>     ;jne    short set_extd_audio_status_1
                                <1>     ;and    al, 0FEh ; disable VRA bit (set sample rate to 48000 Hz)
                                <1>     ;jmp    short set_extd_audio_status_2
                                <1> ;set_extd_audio_status_1:
00013D55 BA2A000000             <1>     mov     edx, CODEC_EXT_AUDIO_CTRL_REG
00013D5A E86EFEFFFF             <1>     call    codec_write
                                <1>     ;jc     short cconfig_error
                                <1>
                                <1> set_sample_rate:
                                <1>     ;movzx  eax, word [audio_freq]
00013D5F 66A1[E6880100]         <1>     mov     ax, [audio_freq]
00013D65 BA2C000000             <1>     mov     edx, CODEC_PCM_FRONT_DACRATE_REG ; 2Ch ; PCM Front DAC Rate
                                <1> ;set_extd_audio_status_2:
00013D6A E85EFEFFFF             <1>     call    codec_write
                                <1>
                                <1>     ; 01/09/2020
```

```
                                <1>     ; set AD1980 MCB register (Index 76h) to 0C00h
                                <1>     ; (CLDIS, HPSEL)
                                <1>     ;mov    ax, 0C00h
                                <1>     ;mov    edx, CODEC_MISC_CRTL_BITS_REG ; 76h
                                <1>     ;                       ; Miscellaneous Control Bit Register
                                <1>     ;call   codec_write
                                <1>     ;
                                <1>
00013D6F B8[F0880100]           <1>         mov eax, audio_bdl_buff
                                <1>
                                <1>     ; 12/11/2016 - Erdogan Tan
                                <1>     ; (Ref: KolibriOS, vt823x.asm, 'create_primary_buff')
                                <1>     ;mov    edx, VIADEV_PLAYBACK + VIA_REG_OFFSET_TABLE_PTR
                                <1>         ; 06/08/2022
00013D74 29D2                   <1>     sub     edx, edx
                                <1>     ;mov    dl, VIADEV_PLAYBACK + VIA_REG_OFFSET_TABLE_PTR
                                <1>     ; edx = 0
00013D76 E81DFEFFFF             <1>     call    ctrl_io_w32
                                <1>
                                <1>     ;call   codec_check_ready
                                <1>
00013D7B 66BA0200               <1>     mov     dx, VIADEV_PLAYBACK + VIA_REG_OFS_PLAYBACK_VOLUME_L
                                <1>         ;mov       eax, 2  ; 31
00013D7F B01F                   <1>     mov     al, 31
00013D81 2A05[EC880100]         <1>         sub al, [audio_master_volume_l]
00013D87 E8FAFDFFFF             <1>     call    ctrl_io_w8
                                <1>
                                <1>     ;call   codec_check_ready
                                <1>
00013D8C 66BA0300               <1>         mov     dx, VIADEV_PLAYBACK + VIA_REG_OFS_PLAYBACK_VOLUME_R
                                <1>         ;mov       ax, 2   ; 31
00013D90 B01F                   <1>     mov     al, 31
00013D92 2A05[ED880100]         <1>         sub al, [audio_master_volume_r]
00013D98 E8E9FDFFFF             <1>     call    ctrl_io_w8
                                <1>
                                <1>     ;call   codec_check_ready
                                <1> ;
                                <1> ;
                                <1> ; All set. Let's play some music.
                                <1> ;
                                <1> ;
                                <1>             ;mov    dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STOP_IDX
                                <1>         ;mov    ax, VIA8233_REG_TYPE_16BIT or VIA8233_REG_TYPE_STEREO or 0xfffff or 0xff000000
                                <1>         ;call   ctrl_io_w32
                                <1>
                                <1>     ;call   codec_check_ready
                                <1>
                                <1>     ; 08/12/2016
                                <1>     ; 07/10/2016
                                <1>         ;;mov    al, 1
                                <1>         ;mov       al, 31
                                <1>     ; 22/07/2020
00013D9D B0FF                   <1>     mov     al, 0FFh
00013D9F E813000000             <1>     call    set_VT8233_LastValidIndex
                                <1>
00013DA4 C605[EA880100]01       <1>     mov     byte [audio_play_cmd], 1 ; play command (do not stop) !
                                <1>
                                <1>     ; 22/07/2020
                                <1>     ;mov    byte [audio_flag], 0  ; clear half buffer flag
                                <1>
                                <1> vt8233_play: ; continue to play
                                <1>     ; 22/04/2017
                                <1>     ;mov    al, VIA_REG_CTRL_INT
                                <1>             ;or     al, VIA_REG_CTRL_START
                                <1>         ;;mov      al, VIA_REG_CTRL_AUTOSTART + VIA_REG_CTRL_START
                                <1>     ; 22/07/2020
00013DAB B0A1                   <1>     mov     al, VIA_REG_CTRL_AUTOSTART + VIA_REG_CTRL_START + VIA_REG_CTRL_INT_FLAG
                                <1>
00013DAD 66BA0100               <1>     mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
00013DB1 E8D0FDFFFF             <1>         call    ctrl_io_w8
                                <1>     ;call   codec_check_ready
                                <1>     ;retn
                                <1>     ;jmp    codec_check_ready
00013DB6 C3                     <1>     retn
                                <1>
                                <1> ;input AL = index # to stop on
                                <1> set_VT8233_LastValidIndex:
                                <1>     ; 06/08/2022 - TRDOS 386 Kernel v2.0.5
                                <1>     ; 23/07/2020
                                <1>     ; 10/06/2017
                                <1>     ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
                                <1>     ; 24/03/2017 - 'PLAYER.COM' ('via_wav.asm' - 29/11/2016)
                                <1>     ; 19/11/2016
                                <1>     ; 14/11/2016 - Erdogan Tan (Ref: VIA VT8235.PDF, Page 110)
                                <1>     ; 12/11/2016 - Erdogan Tan
                                <1>     ; (Ref: KolibriOS, vt823x.asm, 'create_primary_buff')
                                <1>     ;push   edx
                                <1>     ;push   ax
00013DB7 50                     <1>     push    eax ; 23/07/2020
                                <1>     ;push   ecx
00013DB8 0FB705[E6880100]       <1>     movzx   eax, word [audio_freq] ; Hertz
00013DBF BA00001000             <1>     mov     edx, 100000h ; 2^20 = 1048576
00013DC4 F7E2                   <1>     mul     edx
00013DC6 B980BB0000             <1>     mov     ecx, 48000
00013DCB F7F1                   <1>     div     ecx
                                <1>     ;and    eax, 0FFFFFh
                                <1>     ;pop    ecx
                                <1>     ;pop    dx
00013DCD 5A                     <1>     pop     edx ; 23/07/2020
00013DCE C1E218                 <1>     shl     edx, 24  ; STOP Index Setting: Bit 24 to 31
00013DD1 09D0                   <1>     or      eax, edx
                                <1>     ; 19/11/2016
00013DD3 803D[E4880100]10       <1>     cmp     byte [audio_bps], 16
00013DDA 7505                   <1>     jne     short sLVI_1
00013DDC 0D00002000             <1>     or      eax, VIA8233_REG_TYPE_16BIT
                                <1> sLVI_1:
00013DE1 803D[E5880100]02       <1>     cmp     byte [audio_stmo], 2
00013DE8 7505                   <1>     jne     short sLVI_2
00013DEA 0D00001000             <1>     or      eax, VIA8233_REG_TYPE_STEREO
                                <1> sLVI_2:
                                <1>     ;mov    edx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STOP_IDX
                                <1>     ; 06/08/2022
00013DEF 29D2                   <1>     sub     edx, edx
00013DF1 B208                   <1>     mov     dl, VIADEV_PLAYBACK + VIA_REG_OFFSET_STOP_IDX
00013DF3 E8A0FDFFFF             <1>     call    ctrl_io_w32
                                <1>     ;call   codec_check_ready
                                <1>     ;pop    edx
00013DF8 C3                     <1>     retn
                                <1>
                                <1> vt8233_pause: ; pause
                                <1>     ; 10/06/2017
                                <1>     ; 22/04/2017
                                <1>     ;mov    al, VIA_REG_CTRL_INT
                                <1>         ;or al, VIA_REG_CTRL_PAUSE
```

```
                                <1>     ; 23/07/2020
00013DF9 B029                   <1>     mov     al, VIA_REG_CTRL_PAUSE+VIA_REG_CTRL_INT_FLAG+VIA_REG_CTRL_AUTOSTART
                                <1>
00013DFB 66BA0100               <1>     mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
00013DFF E882FDFFFF             <1>         call    ctrl_io_w8
                                <1>     ;call   codec_check_ready
                                <1>     ;retn
                                <1>     ;jmp    codec_check_ready
00013E04 C3                     <1>     retn
                                <1>
                                <1> vt8233_reset:
                                <1>     ; 22/04/2017
                                <1>     ; reset VT8237R (vt8233) Audio Controller
                                <1>     ;cmp    byte [audio_play_cmd], 1
                                <1>     ;jna    short vt8233_rst_0
00013E05 C605[EA880100]00       <1>     mov     byte [audio_play_cmd], 0 ; stop !
                                <1> vt8233_rst_0:
00013E0C E8DAFCFFFF             <1>     call    reset_codec
00013E11 720A                   <1>     jc      short vt8233_rst_1 ; codec error !
                                <1>     ; eax = 1
00013E13 E852FDFFFF             <1>     call    codec_io_w16 ; w32
00013E18 E8B9FEFFFF             <1>     call    channel_reset
                                <1> vt8233_rst_1:
                                <1> vt8233_vol_1:       ; 06/08/2022
00013E1D C3                     <1>     retn
                                <1>
                                <1> vt8233_volume:
                                <1>     ; set VT8237R (vt8233) sound volume level
                                <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
                                <1>     ; 24/04/2017
                                <1>     ; 22/04/2017
                                <1>     ; bl = component (0 = master/playback/lineout volume)
                                <1>     ; cl = left channel volume level (0 to 31)
                                <1>     ; ch = right channel volume level (0 to 31)
                                <1>
00013E1E 08DB                   <1>     or      bl, bl
00013E20 75FB                   <1>     jnz     short vt8233_vol_1 ; temporary !
00013E22 66B81F1F               <1>     mov     ax, 1F1Fh ; 31,31
00013E26 38C1                   <1>     cmp     cl, al
00013E28 77F3                   <1>     ja      short vt8233_vol_1 ; temporary !
00013E2A 38E5                   <1>     cmp     ch, ah
00013E2C 77EF                   <1>     ja      short vt8233_vol_1 ; temporary !
00013E2E 66890D[EC880100]       <1>     mov     [audio_master_volume], cx
00013E35 6629C8                 <1>     sub     ax, cx
                                <1>     ;mov    edx, CODEC_MASTER_VOL_REG ; 02h ; Line Out
                                <1>     ; 06/08/2022
00013E38 29D2                   <1>     sub     edx, edx
00013E3A B202                   <1>     mov     dl, CODEC_MASTER_VOL_REG ; 02h ; Line Out
                                <1>     ; 06/08/2022
00013E3C E98CFDFFFF             <1>     jmp     codec_write
                                <1>     ;call   codec_write
                                <1> ;vt8233_vol_1:
                                <1>     ;retn
                                <1>
                                <1> ; CODE for SOUND BLASTER 16
                                <1>
                                <1> DetectSB:
                                <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
                                <1>     ; 24/04/2017
                                <1>     ;pushad
                                <1> ScanPort:
                                <1>     ; 06/08/2022
00013E41 66BB1002               <1>     mov     bx, 0210h       ; start scanning ports
                                <1>                             ; 210h, 220h, .. 260h
                                <1>     ; 06/08/2022
00013E45 31C9                   <1>     xor     ecx, ecx
00013E47 88FE                   <1>     mov     dh, bh
                                <1> ResetDSP:
                                <1>     ;mov    dx, bx          ; try to reset the DSP.
                                <1>     ;add    dx, 06h
                                <1>     ; 06/08/2022
00013E49 88DA                   <1>     mov     dl, bl
00013E4B 80C206                 <1>     add     dl, 06h
                                <1>
00013E4E B001                   <1>     mov     al, 1
00013E50 EE                     <1>     out     dx, al
                                <1>
00013E51 EC                     <1>     in      al, dx
00013E52 EC                     <1>     in      al, dx
00013E53 EC                     <1>     in      al, dx
00013E54 EC                     <1>     in      al, dx
                                <1>
00013E55 30C0                   <1>     xor     al, al
00013E57 EE                     <1>     out     dx, al
                                <1>
                                <1>     ;add    dx, 08h
                                <1>     ; 06/08/2022
00013E58 80C208                 <1>     add     dl, 08h
                                <1>     ;mov    cx, 100
00013E5B B164                   <1>     mov     cl, 100
                                <1> WaitID:
00013E5D EC                     <1>     in      al, dx
00013E5E 08C0                   <1>     or      al, al
00013E60 7804                   <1>     js      short GetID
00013E62 E2F9                   <1>     loop    WaitID
00013E64 EB0D                   <1>     jmp     short NextPort
                                <1> GetID:
                                <1>     ;sub    dx, 04h
                                <1>     ; 06/08/2022
00013E66 80EA04                 <1>     sub     dl, 04h
00013E69 EC                     <1>     in      al, dx
00013E6A 3CAA                   <1>     cmp     al, 0AAh
00013E6C 740F                   <1>     je      short Found
                                <1>     ;add    dx, 04h
                                <1>     ; 06/08/2022
00013E6E 80C204                 <1>     add     dl, 04h
00013E71 E2EA                   <1>     loop    WaitID
                                <1> NextPort:
                                <1>     ;add    bx, 10h         ; if not response,
                                <1>     ; 06/08/2022
00013E73 80C310                 <1>     add     bl, 10h
                                <1>     ;cmp    bx, 260h        ; try the next port.
00013E76 80FB60                 <1>     cmp     bl, 60h
00013E79 76CE                   <1>     jbe     short ResetDSP
00013E7B F9                     <1>     stc
00013E7C C3                     <1>     retn
                                <1> Found:
00013E7D 66891D[B6880100]       <1>     mov     [audio_io_base], bx   ; SB Port Address Found!
                                <1> ScanIRQ:
                                <1> SetIrqs:
00013E84 28C0                   <1>     sub     al, al ; 0
00013E86 A2[AC880100]           <1>     mov     [IRQnum], al ; reset
00013E8B A2[B2880100]           <1>     mov     [audio_intr], al ; reset
                                <1>
```

```
1835                                    <1>     ; ah > 0 -> set IRQ vector
1836                                    <1>     ; al = IRQ number
1837                                    <1>     ;mov    ax, 103h ; IRQ 3
1838                                    <1>     ;call   set_hardware_int_vector
1839                                    <1>     ;mov    ax, 104h ; IRQ 4
1840                                    <1>     ;call   set_hardware_int_vector
1841 00013E90 66B80501                 <1>     mov     ax, 105h ; IRQ 5
1842 00013E94 E852DFFFFF               <1>     call    set_hardware_int_vector
1843 00013E99 66B80701                 <1>     mov     ax, 107h ; IRQ 7
1844 00013E9D E849DFFFFF               <1>     call    set_hardware_int_vector
1845                                    <1>
1846 00013EA2 668B15[B6880100]         <1>     mov     dx, [audio_io_base] ; tells to the SB to
1847                                    <1>     ;add    dx, 0Ch             ; generate a IRQ!
1848                                    <1>     ; 06/08/2022
1849 00013EA9 80C20C                   <1>     add     dl, 0Ch
1850                                    <1> WaitSb:
1851 00013EAC EC                       <1>     in      al, dx
1852 00013EAD 08C0                     <1>     or      al, al
1853 00013EAF 78FB                     <1>     js      short WaitSb
1854 00013EB1 B0F2                     <1>     mov     al, 0F2h
1855 00013EB3 EE                       <1>     out     dx, al
1856                                    <1>
1857 00013EB4 31C9                     <1>     xor     ecx, ecx      ; wait until IRQ level
1858                                    <1> WaitIRQ:
1859 00013EB6 A0[AC880100]             <1>     mov     al, [IRQnum]
1860 00013EBB 3C00                     <1>     cmp     al, 0 ; is changed or timeout.
1861 00013EBD 7706                     <1>     ja      short IrqOk
1862 00013EBF 6649                     <1>     dec     cx
1863 00013EC1 75F3                     <1>     jnz     short WaitIRQ
1864 00013EC3 EB14                     <1>     jmp     short RestoreIrqs
1865                                    <1> IrqOk:
1866 00013EC5 A2[B2880100]             <1>     mov     [audio_intr], al ; set
1867 00013ECA 668B15[B6880100]         <1>     mov     dx, [audio_io_base]
1868                                    <1>     ;add    dx, 0Eh
1869                                    <1>     ; 06/08/2022
1870 00013ED1 80C20E                   <1>     add     dl, 0Eh
1871 00013ED4 EC                       <1>     in      al, dx ; SB acknowledge.
1872 00013ED5 B020                     <1>     mov     al, 20h
1873 00013ED7 E620                     <1>     out     20h, al ; Hardware acknowledge.
1874                                    <1>
1875                                    <1> RestoreIrqs:
1876                                    <1>     ; ah = 0 -> reset IRQ vector
1877                                    <1>     ; al = IRQ number
1878                                    <1>     ;mov    ax, 3 ; IRQ 3
1879                                    <1>     ;call   set_hardware_int_vector
1880                                    <1>     ;mov    ax, 4 ; IRQ 4
1881                                    <1>     ;call   set_hardware_int_vector
1882 00013ED9 66B80500                 <1>     mov     ax, 5 ; IRQ 5
1883 00013EDD E809DFFFFF               <1>     call    set_hardware_int_vector
1884 00013EE2 66B80700                 <1>     mov     ax, 7 ; IRQ 7
1885 00013EE6 E800DFFFFF               <1>     call    set_hardware_int_vector
1886                                    <1>
1887 00013EEB 31D2                     <1>     xor     edx, edx
1888 00013EED 8915[B8880100]           <1>     mov     [audio_dev_id], edx ; 0
1889 00013EF3 8915[BC880100]           <1>     mov     [audio_vendor], edx ; 0
1890                                    <1>     ; 22/11/2023
1891                                    <1>     ;mov    [audio_stats_cmd], edx ; 0
1892                                    <1>
1893                                    <1>     ;popad
1894                                    <1>
1895 00013EF9 803D[B2880100]01         <1>     cmp     byte [audio_intr], 1 ; IRQ level was changed?
1896                                    <1>
1897 00013F00 C3                       <1>     retn
1898                                    <1>
1899                                    <1> %macro      SbOut  1
1900                                    <1> %%Wait:
1901                                    <1>     in      al, dx
1902                                    <1>     or      al, al
1903                                    <1>     js      short %%Wait
1904                                    <1>     mov     al, %1
1905                                    <1>     out     dx, al
1906                                    <1> %endmacro
1907                                    <1>
1908                                    <1> SbInit_play:
1909                                    <1>     ; 28/01/2025 - TRDOS 386 Kernel v2.0.10
1910                                    <1>     ; 06/08/2022 - TRDOS 386 Kernel v2.0.5
1911                                    <1>     ; 22/10/2017
1912                                    <1>     ; 20/10/2017
1913                                    <1>     ; 06/10/2017
1914                                    <1>     ; 13/07/2017 - 09/08/2017
1915                                    <1>     ; 24/04/2017 - 15/05/2017 - 24/06/2017
1916                                    <1>     ;pushad
1917                                    <1> SetBuffer:
1918                                    <1>     ;mov    byte [DmaFlag], 0
1919                                    <1>
1920 00013F01 8B1D[D0880100]           <1>     mov     ebx, [audio_dma_buff] ; physical addr of DMA buff
1921 00013F07 89DF                     <1>     mov     edi, ebx
1922                                    <1>     ;mov    ecx, [audio_dmabuff_size]
1923                                    <1>     ; 28/01/2025 (BugFix)
1924 00013F09 8B0D[D8880100]           <1>     mov     ecx, [dma_hbuff_size]
1925                                    <1>     ;shl    ecx, 1 ; * 2 ; *!*
1926                                    <1>
1927 00013F0F 803D[E4880100]10         <1>     cmp     byte [audio_bps], 16
1928 00013F16 752F                     <1>     jne     short sbInit_0 ; set 8 bit DMA buffer
1929                                    <1>
1930                                    <1>     ; 09/08/2017
1931                                    <1>     ; convert byte count to word count
1932                                    <1>     ; 28/01/2025
1933                                    <1>     ;shr    ecx, 1 ; *!*
1934                                    <1>
1935 00013F18 49                       <1>     dec     ecx ; word count - 1
1936                                    <1>     ; convert byte offset to word offset
1937 00013F19 D1EB                     <1>     shr     ebx, 1
1938                                    <1>
1939                                    <1>     ; 16 bit DMA buffer setting (DMA channel 5)
1940 00013F1B B005                     <1>     mov     al, 05h  ; set mask bit for channel 5  (4+1)
1941 00013F1D E6D4                     <1>     out     0D4h, al
1942                                    <1>
1943 00013F1F 30C0                     <1>     xor     al, al   ; stops all DMA processes on selected channel
1944 00013F21 E6D8                     <1>     out     0D8h, al ; clear selected channel register
1945                                    <1>
1946 00013F23 88D8                     <1>     mov     al, bl  ; byte 0 of DMA buffer offset in words (physical)
1947 00013F25 E6C4                     <1>     out     0C4h, al ; DMA channel 5 port number
1948                                    <1>
1949 00013F27 88F8                     <1>     mov     al, bh   ; byte 1 of DMA buffer offset in words (physical)
1950 00013F29 E6C4                     <1>     out     0C4h, al
1951                                    <1>
1952                                    <1>     ; 09/08/2017
1953 00013F2B C1EB0F                   <1>     shr     ebx, 15  ; complete 16 bit shift
1954 00013F2E 80E3FE                   <1>     and     bl, 0FEh ; clear bit 0 (not necessary, it will be ignored)
1955                                    <1>
1956 00013F31 88D8                     <1>     mov     al, bl   ; byte 2 of DMA buffer address (physical)
1957 00013F33 E68B                     <1>     out     8Bh, al  ; page register port addr for channel 5 ; 13/07/2017
1958                                    <1>
```

```
1959 00013F35 88C8                <1>     mov     al, cl   ; low byte of DMA count - 1
1960 00013F37 E6C6                <1>     out     0C6h, al ; count register port addr for channel 1
1961                              <1>
1962 00013F39 88E8                <1>     mov     al, ch   ; high byte of DMA count - 1
1963 00013F3B E6C6                <1>     out     0C6h, al
1964                              <1>
1965                              <1>     ; channel 5, read, autoinitialized, single mode
1966                              <1>     ;mov    al, 49h
1967 00013F3D B059                <1>     mov     al, 59h  ; 06/10/2017
1968 00013F3F E6D6                <1>     out     0D6h, al ; DMA mode register port address
1969                              <1>
1970 00013F41 B001                <1>     mov     al, 01h  ; clear mask bit for channel 1
1971 00013F43 E6D4                <1>     out     0D4h, al ; DMA mask register port address
1972                              <1>
1973 00013F45 EB2A                <1>     jmp     short ClearBuffer
1974                              <1>
1975                              <1> sbInit_0:
1976                              <1>     ; 28/01/2025
1977 00013F47 D1E1                <1>     shl     ecx, 1 ; half buffer size * 2 ; *!*
1978                              <1>
1979 00013F49 49                  <1>     dec     ecx    ; 09/08/2017
1980                              <1>
1981                              <1>     ; 8 bit DMA buffer setting (DMA channel 1)
1982 00013F4A B005                <1>     mov     al, 05h ; set mask bit for channel 1  (4+1)
1983 00013F4C E60A                <1>     out     0Ah, al ; DMA mask register
1984                              <1>
1985 00013F4E 30C0                <1>     xor     al, al  ; stops all DMA processes on selected channel
1986 00013F50 E60C                <1>     out     0Ch, al ; clear selected channel register
1987                              <1>
1988 00013F52 88D8                <1>     mov     al, bl ; byte 0 of DMA buffer address (physical)
1989 00013F54 E602                <1>     out     02h, al ; DMA channel 1 port number
1990                              <1>
1991 00013F56 88F8                <1>     mov     al, bh  ; byte 1 of DMA buffer address (physical)
1992 00013F58 E602                <1>     out     02h, al
1993                              <1>
1994 00013F5A C1EB10              <1>     shr     ebx, 16
1995                              <1>
1996 00013F5D 88D8                <1>     mov     al, bl  ; byte 2 of DMA buffer address (physical)
1997 00013F5F E683                <1>     out     83h, al ; page register port addr for channel 1
1998                              <1>
1999 00013F61 88C8                <1>     mov     al, cl  ; low byte of DMA count - 1
2000 00013F63 E603                <1>     out     03h, al ; count register port addr for channel 1
2001                              <1>
2002 00013F65 88E8                <1>     mov     al, ch  ; high byte of DMA count - 1
2003 00013F67 E603                <1>     out     03h, al
2004                              <1>
2005                              <1>     ; channel 1, read, autoinitialized, single mode
2006                              <1>     ;mov    al, 49h
2007 00013F69 B059                <1>     mov     al, 59h ; 06/10/2017
2008 00013F6B E60B                <1>     out     0Bh, al ; DMA mode register port address
2009                              <1>
2010 00013F6D B001                <1>     mov     al, 01h ; clear mask bit for channel 1
2011 00013F6F E60A                <1>     out     0Ah, al ; DMA mask register port address
2012                              <1>
2013                              <1> ClearBuffer:
2014                              <1>     ;;mov   edi, [audio_dma_buff]
2015                              <1>     ;;mov   ecx, [audio_dmabuff_size]
2016                              <1>     ;inc    ecx
2017                              <1>     ;mov    al, 80h
2018                              <1>     ;;cld
2019                              <1>     ;rep    stosb
2020                              <1> SetIrq:
2021                              <1>     ;mov    ebx, SbIrqhandler
2022                              <1>     ;mov    al, [audio_intr] ; IRQ number
2023                              <1>     ;call   set_dev_IRQ_service
2024                              <1>     ;; SETUP (audio) INTERRUPT CALLBACK SERVICE
2025                              <1>     ;mov    bl, [audio_intr] ; IRQ number
2026                              <1>     ;mov    bh, [audio_cb_mode]
2027                              <1>     ;inc    bh  ; 1 = Signal Response Byte method (fixed value)
2028                              <1>     ;           ; 2 = Callback service method
2029                              <1>     ;           ; 3 = Auto Increment S.R.B. method
2030                              <1>     ;mov    cl, [audio_srb]
2031                              <1>     ;mov    edx, [audio_cb_addr]
2032                              <1>     ;mov    al, [audio_user]
2033                              <1>     ;call   set_irq_callback_service
2034                              <1> ResetDsp:
2035 00013F71 668B15[B6880100]    <1>     mov     dx, [audio_io_base]
2036                              <1>     ;add    dx, 06h
2037                              <1>     ; 06/08/2022
2038 00013F78 80C206              <1>     add     dl, 06h
2039 00013F7B B001                <1>     mov     al, 1
2040 00013F7D EE                  <1>     out     dx, al
2041                              <1>
2042 00013F7E EC                  <1>     in      al, dx
2043 00013F7F EC                  <1>     in      al, dx
2044 00013F80 EC                  <1>     in      al, dx
2045 00013F81 EC                  <1>     in      al, dx
2046                              <1>
2047 00013F82 30C0                <1>     xor     al, al
2048 00013F84 EE                  <1>     out     dx, al
2049                              <1>
2050                              <1>     ;mov    cx, 100
2051                              <1>     ; 06/08/2022
2052 00013F85 29C9                <1>     sub     ecx, ecx
2053 00013F87 B164                <1>     mov     cl, 100
2054 00013F89 28E4                <1>     sub     ah, ah ; 0
2055                              <1> WaitId:
2056 00013F8B 668B15[B6880100]    <1>     mov     dx, [audio_io_base]
2057                              <1>     ;add    dx, 0Eh
2058                              <1>     ; 06/08/2022
2059 00013F92 80C20E              <1>     add     dl, 0Eh
2060 00013F95 EC                  <1>     in      al, dx
2061 00013F96 08C0                <1>     or      al, al
2062 00013F98 7807                <1>     js      short sb_GetId
2063 00013F9A E2EF                <1>     loop    WaitId
2064 00013F9C E9B1000000          <1>     jmp     sb_Exit
2065                              <1> sb_GetId:
2066 00013FA1 668B15[B6880100]    <1>     mov     dx, [audio_io_base]
2067                              <1>     ;add    dx, 0Ah
2068                              <1>     ; 06/08/2022
2069 00013FA8 80C20A              <1>     add     dl, 0Ah
2070 00013FAB EC                  <1>     in      al, dx
2071 00013FAC 3CAA                <1>     cmp     al, 0AAh
2072 00013FAE 7407                <1>     je      short SbOk
2073 00013FB0 E2D9                <1>     loop    WaitId
2074 00013FB2 E99B000000          <1>     jmp     sb_Exit
2075                              <1> SbOk:
2076 00013FB7 668B15[B6880100]    <1>     mov     dx, [audio_io_base]
2077                              <1>     ;add    dx, 0Ch
2078                              <1>     ; 06/08/2022
2079 00013FBE 80C20C              <1>     add     dl, 0Ch
2080                              <1>     SbOut   0D1h ; Turn on speaker
1900                              <2> %%Wait:
1901 00013FC1 EC                  <2>  in al, dx
```

```
1902 00013FC2 08C0                <2>  or al, al
1903 00013FC4 78FB                <2>  js short %%Wait
1904 00013FC6 B0D1                <2>  mov al, %1
1905 00013FC8 EE                  <2>  out dx, al
2081                              <1>     SbOut   41h ; 8h bit or 16 bit transfer
1900                              <2> %%Wait:
1901 00013FC9 EC                  <2>  in al, dx
1902 00013FCA 08C0                <2>  or al, al
1903 00013FCC 78FB                <2>  js short %%Wait
1904 00013FCE B041                <2>  mov al, %1
1905 00013FD0 EE                  <2>  out dx, al
2082 00013FD1 668B1D[E6880100]    <1>     mov    bx, [audio_freq] ; sampling rate (Hz)
2083                              <1>     SbOut  bh ; sampling rate high byte
1900                              <2> %%Wait:
1901 00013FD8 EC                  <2>  in al, dx
1902 00013FD9 08C0                <2>  or al, al
1903 00013FDB 78FB                <2>  js short %%Wait
1904 00013FDD 88F8                <2>  mov al, %1
1905 00013FDF EE                  <2>  out dx, al
2084                              <1>     SbOut  bl ; sampling rate low byte
1900                              <2> %%Wait:
1901 00013FE0 EC                  <2>  in al, dx
1902 00013FE1 08C0                <2>  or al, al
1903 00013FE3 78FB                <2>  js short %%Wait
1904 00013FE5 88D8                <2>  mov al, %1
1905 00013FE7 EE                  <2>  out dx, al
2085                              <1>
2086                              <1>     ; 22/05/2017
2087 00013FE8 E8BD000000          <1>     call   sb16_volume_initial ; 15/05/2017
2088                              <1>     ; 20/05/2017
2089                              <1>     ;call  sb16_volume
2090                              <1>
2091                              <1> StartDma:
2092                              <1>     ; autoinitialized mode
2093 00013FED 803D[E4880100]10    <1>     cmp    byte [audio_bps], 16 ; 16 bit samples
2094 00013FF4 7411                <1>     je     short sb_play_1
2095                              <1>     ; 8 bit samples
2096 00013FF6 66BBC600            <1>     mov    bx, 0C6h ; 8 bit output (0C6h)
2097 00013FFA 803D[E5880100]02    <1>     cmp    byte [audio_stmo], 2 ; 1 = mono, 2 = stereo
2098 00014001 7214                <1>     jb     short sb_play_2
2099 00014003 B720                <1>     mov    bh, 20h ; 8 bit stereo (20h)
2100 00014005 EB10                <1>     jmp    short sb_play_2
2101                              <1> sb_play_1:
2102                              <1>     ; 16 bit samples
2103 00014007 66BBB610            <1>     mov    bx, 10B6h ; 16 bit output (0B6h)
2104 0001400B 803D[E5880100]02    <1>     cmp    byte [audio_stmo], 2 ; 1 = mono, 2 = stereo
2105 00014012 7203                <1>     jb     short sb_play_2
2106 00014014 80C720              <1>     add    bh, 20h ; 16 bit stereo (30h)
2107                              <1> sb_play_2:
2108                              <1>     ; PCM output (8/16 bit mono autoinitialized transfer)
2109                              <1>     SbOut   bl ; bCommand
1900                              <2> %%Wait:
1901 00014017 EC                  <2>  in al, dx
1902 00014018 08C0                <2>  or al, al
1903 0001401A 78FB                <2>  js short %%Wait
1904 0001401C 88D8                <2>  mov al, %1
1905 0001401E EE                  <2>  out dx, al
2110                              <1>     SbOut  bh ; bMode
1900                              <2> %%Wait:
1901 0001401F EC                  <2>  in al, dx
1902 00014020 08C0                <2>  or al, al
1903 00014022 78FB                <2>  js short %%Wait
1904 00014024 88F8                <2>  mov al, %1
1905 00014026 EE                  <2>  out dx, al
2111 00014027 8B1D[D4880100]      <1>     mov    ebx, [audio_dmabuff_size]  ; 15/05/2017
2112 0001402D D1EB                <1>     shr    ebx, 1 ; half buffer size
2113                              <1>     ; 20/10/2017
2114 0001402F 803D[E4880100]10    <1>     cmp    byte [audio_bps], 16 ; 16 bit DMA
2115 00014036 7502                <1>     jne    short sb_play_3
2116 00014038 D1EB                <1>     shr    ebx, 1 ; byte count to word count
2117                              <1> sb_play_3:
2118                              <1>     ;dec   bx  ; wBlkSize is one less than the actual size
2119                              <1>     ; 06/08/2022
2120 0001403A 4B                  <1>     dec    ebx
2121                              <1>     SbOut   bl
1900                              <2> %%Wait:
1901 0001403B EC                  <2>  in al, dx
1902 0001403C 08C0                <2>  or al, al
1903 0001403E 78FB                <2>  js short %%Wait
1904 00014040 88D8                <2>  mov al, %1
1905 00014042 EE                  <2>  out dx, al
2122                              <1>     SbOut   bh
1900                              <2> %%Wait:
1901 00014043 EC                  <2>  in al, dx
1902 00014044 08C0                <2>  or al, al
1903 00014046 78FB                <2>  js short %%Wait
1904 00014048 88F8                <2>  mov al, %1
1905 0001404A EE                  <2>  out dx, al
2123                              <1>
2124 0001404B C605[EA880100]01    <1>     mov    byte [audio_play_cmd], 1 ; playing !
2125                              <1>
2126                              <1>     ;; Set Voice and master volumes
2127                              <1>     ;mov   dx, [audio_io_base]
2128                              <1>     ;add   dl, 4 ; Mixer chip Register Address Port
2129                              <1>     ;SbOut 30h   ; select Master Volume Register (L)
2130                              <1>     ;inc   dl    ; Mixer chip Register Data Port
2131                              <1>     ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
2132                              <1>     ;dec   dl
2133                              <1>     ;SbOut 31h   ; select Master Volume Register (R)
2134                              <1>     ;inc   dl
2135                              <1>     ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
2136                              <1>     ;dec   dl
2137                              <1>     ;SbOut 32h   ; select Voice Volume Register (L)
2138                              <1>     ;inc   dl
2139                              <1>     ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
2140                              <1>     ;dec   dl
2141                              <1>     ;SbOut 33h   ; select Voice Volume Register (R)
2142                              <1>     ;inc   dl
2143                              <1>     ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
2144                              <1>     ;;
2145                              <1>     ;dec   dl
2146                              <1>     ;SbOut 44h   ; select Treble Register (L)
2147                              <1>     ;inc   dl
2148                              <1>     ;SbOut 0F0h  ; Max. Treble value is 15 (15*16)
2149                              <1>     ;dec   dl
2150                              <1>     ;SbOut 45h   ; select Treble Register (R)
2151                              <1>     ;inc   dl
2152                              <1>     ;SbOut 0F0h  ; Max. Treble value is 15 (15*16)
2153                              <1>     ;dec   dl
2154                              <1>     ;SbOut 46h   ; select Bass Register (L)
2155                              <1>     ;inc   dl
2156                              <1>     ;SbOut 0F0h  ; Max. Bass value is 15 (15*16)
2157                              <1>     ;dec   dl
2158                              <1>     ;SbOut 47h   ; select Bass Register (R)
```

```
                                <1>     ;inc    dl
                                <1>     ;SbOut  0F0h  ; Max. Bass value is 15 (15*16)
                                <1>
                                <1> sb_Exit:
                                <1>     ;popad
00014052 C3                     <1>     retn
                                <1>
                                <1> sb16_int_handler:
                                <1>     ; Interrupt Handler for Sound Blaster 16 Audio Card
                                <1>     ; Note: called by 'dev_IRQ_service'
                                <1>     ; 28/01/2025
                                <1>     ;       TRDOS 386 Kernel v2.0.10
                                <1>     ; 20/10/2017
                                <1>     ; 12/10/2017
                                <1>     ; 10/10/2017
                                <1>     ; 12/05/2017, 09/10/2017
                                <1>     ; 24/04/2017 (TRDOS 386 kernel, 'audio.s')
                                <1>     ; 10/03/2017 - 'PLAYWAV.PRG' ('playwav.s')
                                <1>
                                <1>     ;push   eax ; * must be saved !
                                <1>     ;push   ebx ; * must be saved !
                                <1>     ;push   ecx
                                <1>     ;push   edx
                                <1>     ;push   esi
                                <1>     ;push   edi
                                <1>
00014053 668B15[B6880100]       <1>     mov     dx, [audio_io_base]
                                <1>     ; 20/10/2017
0001405A 80C20F                 <1>     add     dl, 0Fh ; 2xFh (DSP 16 bit intr ack)
0001405D 803D[E4880100]10       <1>     cmp     byte [audio_bps], 16
00014064 7402                   <1>     je      short sb_irq_16bit_ack
                                <1> sb_irq_8bit_ack:
00014066 FECA                   <1>     dec     dl  ; 2xEh (DSP 8 bit intr ack)
                                <1> sb_irq_16bit_ack:
00014068 EC                     <1>     in      al, dx
                                <1>
                                <1>     ;cmp    byte [audio_busy], 0
                                <1>     ;ja     short sb_irq_h3
                                <1>
                                <1>     ;mov    byte [audio_busy], 1
                                <1>
00014069 803D[EA880100]01       <1>     cmp     byte [audio_play_cmd], 1
00014070 7307                   <1>     jnb     short sb_irq_h1
                                <1> sb_irq_h0:
00014072 E8A1000000             <1>     call    sb16_stop
00014077 EB29                   <1>     jmp     short sb_irq_h3
                                <1> sb_irq_h1:
                                <1>     ;call   sb16_tuneloop
                                <1>     ; 09/10/2017
                                <1> sb16_tuneloop:
00014079 8B3D[D0880100]         <1>     mov     edi, [audio_dma_buff]
                                <1>     ;mov    ecx, [audio_dmabuff_size]
                                <1>     ;shr    ecx, 1 ; dma buff size / 2 = half buffer size
                                <1>     ; 28/01/2025 (BugFix)
0001407F 8B0D[D8880100]         <1>     mov     ecx, [dma_hbuff_size]
                                <1>
                                <1>     ; 22/05/2017
00014085 F605[DC880100]01       <1>     test    byte [audio_flag], 1  ; Current flag value
0001408C 7402                   <1>     jz      short sb_tlp1 ; EOL (Half Buffer 1 must be filled)
                                <1>     ; FLAG (Half Buffer 2 must be filled)
0001408E 01CF                   <1>     add     edi, ecx
                                <1>     ; 15/05/2017
                                <1> sb_tlp1:
00014090 8B35[C8880100]         <1>     mov     esi, [audio_p_buffer] ; phy addr of audio buff
                                <1>     ;rep    movsb
00014096 C1E902                 <1>     shr     ecx, 2 ; half buff size / 4
00014099 F3A5                   <1>     rep     movsd
                                <1>     ;retn
                                <1>
                                <1>     ; 10/10/2017
                                <1>     ; switch flag value
0001409B 8035[DC880100]01       <1>     xor     byte [audio_flag], 1
                                <1>
                                <1>     ; 12/10/2017
                                <1>     ; [audio_flag] = 0 : Playing dma half buffer 2 (odd intr count)
                                <1>                        ; Next buffer (to update) is dma half buff 1
                                <1>     ;              = 1 : Playing dma half buffer 1 (even intr count)
                                <1>                        ; Next buffer (to update) is dma half buff 2
                                <1>
                                <1> sb_irq_h3:
                                <1>     ;mov    byte [audio_busy], 0
                                <1>
                                <1>     ;pop    edi
                                <1>     ;pop    esi
                                <1>     ;pop    edx
                                <1>     ;pop    ecx
                                <1>     ;pop    ebx ; * must be restored !
                                <1>     ;pop    eax ; * must be restored !
                                <1>
000140A2 C3                     <1>     retn
                                <1>
                                <1> sb16_volume:
                                <1>     ; 06/08/2022 (TRDOS 386 v2.0.5)
                                <1>     ; 22/10/2017
                                <1>     ; mov [audio_master_volume_l], cl
                                <1>     ; mov [audio_master_volume_h], ch
000140A3 66890D[EC880100]       <1>     mov     [audio_master_volume], cx
                                <1> sb16_volume_initial:
                                <1>     ;push   dx ; DX (port address) must be saved
                                <1>     ; 06/08/2022
000140AA 52                     <1>     push    edx
000140AB 668B15[B6880100]       <1>     mov     dx, [audio_io_base]
                                <1>     ;add    dx, 4 ; Mixer chip address port
                                <1>     ; 06/08/2022
000140B2 80C204                 <1>     add     dl, 4
000140B5 B022                   <1>     mov     al, 22h ; master volume
000140B7 EE                     <1>     out     dx, al
                                <1>     ;inc    dx
                                <1>     ; 06/08/2022
000140B8 42                     <1>     inc     edx
000140B9 8A25[EC880100]         <1>     mov     ah, [audio_master_volume_l]
000140BF C0EC02                 <1>     shr     ah, 2 ; 32 -> 8 level
000140C2 C0E405                 <1>     shl     ah, 5 ; bit 5 to 7
000140C5 A0[ED880100]           <1>     mov     al, [audio_master_volume_r]
000140CA C0E802                 <1>     shr     al, 2 ; 32 -> 8 level
                                <1>     ;and    al, 0Fh
000140CD D0E0                   <1>     shl     al, 1 ; bit 1 to 3
000140CF 08E0                   <1>     or      al, ah
000140D1 EE                     <1>     out     dx, al
                                <1>     ;pop    dx ; DX (port address) must be restored
                                <1>     ; 06/08/2022
000140D2 5A                     <1>     pop     edx
000140D3 C3                     <1>     retn
                                <1>
```

```
                                <1> sb16_pause:
                                <1>     ; 06/08/2022 (TRDOS 386 v2.0.5)
000140D4 668B15[B6880100]       <1>     mov     dx, [audio_io_base]
                                <1>     ;add    dx, 0Ch ; Command & Data Port
                                <1>     ; 06/08/2022
000140DB 80C20C                 <1>     add     dl, 0Ch
000140DE 803D[E4880100]10       <1>     cmp     byte [audio_bps], 16 ; 16 bit samples
000140E5 7404                   <1>     je      short sb_pause_1
                                <1>     ; 8 bit samples
000140E7 B3D0                   <1>     mov     bl, 0D0h ; 8 bit DMA mode
000140E9 EB02                   <1>     jmp     short sb_pause_2
                                <1> sb_pause_1:
                                <1>     ; 16 bit samples
000140EB B3D5                   <1>     mov     bl, 0D5h ; 16 bit DMA mode
                                <1> sb_pause_2:
                                <1>     SbOut   bl ; bCommand
                                <2> %%Wait:
000140ED EC                     <2>  in al, dx
000140EE 08C0                   <2>  or al, al
000140F0 78FB                   <2>  js short %%Wait
000140F2 88D8                   <2>  mov al, %1
000140F4 EE                     <2>  out dx, al
                                <1> sb_pause_3:
000140F5 C3                     <1>     retn
                                <1>
                                <1> sb16_continue:
                                <1>     ; 06/08/2022 (TRDOS 386 v2.0.5)
000140F6 668B15[B6880100]       <1>     mov     dx, [audio_io_base]
                                <1>     ;add    dx, 0Ch ; Command & Data Port
                                <1>     ; 06/08/2022
000140FD 80C20C                 <1>     add     dl, 0Ch
00014100 803D[E4880100]10       <1>     cmp     byte [audio_bps], 16 ; 16 bit samples
00014107 7404                   <1>     je      short sb_cont_1
                                <1>     ; 8 bit samples
00014109 B3D4                   <1>     mov     bl, 0D4h ; 8 bit DMA mode
0001410B EB02                   <1>     jmp     short sb_cont_2
                                <1> sb_cont_1:
                                <1>     ; 16 bit samples
0001410D B3D6                   <1>     mov     bl, 0D6h ; 16 bit DMA mode
                                <1> sb_cont_2:
                                <1>     SbOut   bl ; bCommand
                                <2> %%Wait:
0001410F EC                     <2>  in al, dx
00014110 08C0                   <2>  or al, al
00014112 78FB                   <2>  js short %%Wait
00014114 88D8                   <2>  mov al, %1
00014116 EE                     <2>  out dx, al
                                <1> sb_cont_3:
00014117 C3                     <1>     retn
                                <1>
                                <1> sb16_stop:
                                <1>     ; 06/08/2022 (TRDOS 386 v2.0.5)
                                <1>     ; 24/04/2017
00014118 803D[EA880100]00       <1>     cmp     byte [audio_play_cmd], 0
0001411F 7647                   <1>     jna     short sb16_stop_4
                                <1>
                                <1>     ; 22/05/2017
00014121 668B15[B6880100]       <1>     mov     dx, [audio_io_base]
                                <1>     ;add    dx, 0Ch
                                <1>     ; 06/08/2022
00014128 80C20C                 <1>     add     dl, 0Ch
                                <1>
0001412B B3D9                   <1>     mov     bl, 0D9h ; exit auto-initialize 16 bit transfer
                                <1>     ; stop  autoinitialized DMA transfer mode
0001412D 803D[E4880100]10       <1>     cmp     byte [audio_bps], 16 ; 16 bit samples
00014134 7402                   <1>     je      short sb16_stop_1
                                <1>     ;mov    bl, 0DAh ; exit auto-initialize 8 bit transfer
00014136 FEC3                   <1>     inc     bl
                                <1> sb16_stop_1:
                                <1>     SbOut   bl ; exit auto-initialize transfer command
                                <2> %%Wait:
00014138 EC                     <2>  in al, dx
00014139 08C0                   <2>  or al, al
0001413B 78FB                   <2>  js short %%Wait
0001413D 88D8                   <2>  mov al, %1
0001413F EE                     <2>  out dx, al
                                <1>
00014140 30C0                   <1>     xor     al, al ; stops all DMA processes on selected channel
                                <1>
00014142 803D[E4880100]10       <1>     cmp     byte [audio_bps], 16 ; 16 bit samples
00014149 7404                   <1>     je      short sb16_stop_2
0001414B E60C                   <1>     out     0Ch, al ; clear selected channel register
0001414D EB02                   <1>     jmp     short sb16_stop_3
                                <1>
                                <1> sb16_stop_2:
0001414F E6D8                   <1>     out     0D8h, al ; clear selected channel register
                                <1>
                                <1> sb16_stop_3:
00014151 C605[EA880100]00       <1>     mov     byte [audio_play_cmd], 0 ; stop !
                                <1> SbDone:
                                <1>     ;mov    dx, [audio_io_base]
                                <1>     ;add    dx, 0Ch
                                <1>     SbOut   0D0h
                                <2> %%Wait:
00014158 EC                     <2>  in al, dx
00014159 08C0                   <2>  or al, al
0001415B 78FB                   <2>  js short %%Wait
0001415D B0D0                   <2>  mov al, %1
0001415F EE                     <2>  out dx, al
                                <1>     SbOut   0D3h
                                <2> %%Wait:
00014160 EC                     <2>  in al, dx
00014161 08C0                   <2>  or al, al
00014163 78FB                   <2>  js short %%Wait
00014165 B0D3                   <2>  mov al, %1
00014167 EE                     <2>  out dx, al
                                <1> sb16_stop_4:
00014168 C3                     <1>     retn
                                <1>
                                <1> sb16_reset:
                                <1>     ; 06/08/2022 (TRDOS 386 v2.0.5)
                                <1>     ; 24/04/2017
00014169 668B15[B6880100]       <1>     mov     dx, [audio_io_base] ; try to reset the DSP.
                                <1>     ;add    dx, 06h
                                <1>     ; 06/08/2022
00014170 80C206                 <1>     add     dl, 06h
00014173 B001                   <1>     mov     al, 1
00014175 EE                     <1>     out     dx, al
                                <1>
00014176 EC                     <1>     in      al, dx
00014177 EC                     <1>     in      al, dx
00014178 EC                     <1>     in      al, dx
00014179 EC                     <1>     in      al, dx
                                <1>
```

```
2377 0001417A 30C0                <1>     xor     al, al
2378 0001417C EE                  <1>     out     dx, al
2379                              <1>
2380                              <1>     ;add    dx, 08h
2381                              <1>     ; 06/08/2022
2382 0001417D 80C208              <1>     add     dl, 08h
2383                              <1>     ;mov    cx, 100
2384 00014180 29C9                <1>     sub     ecx, ecx
2385 00014182 B164                <1>     mov     cl, 100
2386                              <1> sbrstWaitID:
2387 00014184 EC                  <1>     in      al, dx
2388 00014185 08C0                <1>     or      al, al
2389 00014187 7804                <1>     js      short sbrstGetID
2390 00014189 E2F9                <1>     loop    sbrstWaitID
2391 0001418B F9                  <1>     stc
2392 0001418C C3                  <1>     retn
2393                              <1> sbrstGetID:
2394                              <1>     ;sub    dx, 04h
2395                              <1>     ; 06/08/2022
2396 0001418D 80EA04              <1>     sub     dl, 04h
2397 00014190 EC                  <1>     in      al, dx
2398 00014191 3CAA                <1>     cmp     al, 0AAh
2399 00014193 7405                <1>     je      short sb_rst_retn
2400                              <1>     ;add    dx, 04h
2401                              <1>     ; 06/08/2022
2402 00014195 80C204              <1>     add     dl, 04h
2403 00014198 E2EA                <1>     loop    sbrstWaitID
2404                              <1> sb_rst_retn:
2405 0001419A C3                  <1>     retn
2406                              <1>
2407                              <1> ac97_codec_config:
2408                              <1>     ; 06/06/2024
2409                              <1>     ; 04/06/2024
2410                              <1>     ; 03/06/2024
2411                              <1>     ; 02/06/2024
2412                              <1>     ; 01/06/2024 (TRDOS 386 v2.0.8)
2413                              <1>     ; 26/11/2023
2414                              <1>     ; 21/11/2023
2415                              <1>     ; 20/11/2023
2416                              <1>     ; 19/11/2023 (TRDOS 386 v2.0.7)
2417                              <1>     ; 10/06/2017
2418                              <1>     ; 05/06/2017
2419                              <1>     ; 29/05/2017
2420                              <1>     ; 28/05/2017 (TRDOS 386, 'audio.s')
2421                              <1>     ; 07/11/2016 (Erdogan Tan)
2422                              <1>     ; Derived from 'codecConfig' procedure in 'CODEC.ASM'
2423                              <1>     ; .wav player for DOS by Jeff Leyda (02/09/2002)
2424                              <1>
2425                              <1>     ;; 'PLAYER.ASM'
2426                              <1>     ;; get ICH base address regs for mixer and bus master
2427                              <1>
2428                              <1> init_ac97_controller: ; 10/06/2017
2429 0001419B A1[B8880100]        <1>     mov     eax, [audio_dev_id]
2430                              <1>         ;mov       al, NAMBAR_REG
2431                              <1>         ;;call  pciRegRead16                    ; read PCI registers 10-11
2432                              <1>         ;call      pciRegRead32
2433                              <1>     ;and    dx, IO_ADDR_MASK              ; mask off BIT0
2434                              <1>     ;;and   edx, IO_ADDR_MASK
2435                              <1>
2436                              <1>         ;mov       [NAMBAR], dx                 ; save audio mixer base addr
2437                              <1>
2438                              <1>         ;mov       al, NABMBAR_REG
2439                              <1>         ;;call     pciRegRead16
2440                              <1>         ;call      pciRegRead32
2441                              <1>     ;and    dx, 0FFC0h ; IO_ADDR_MASK
2442                              <1>     ;;and   edx, 0FFC0h
2443                              <1>
2444                              <1>         ;mov    [NABMBAR], dx                   ; save bus master base addr
2445                              <1>
2446                              <1>     ;mov    eax, [audio_dev_id]
2447 000141A0 B004                <1>         mov al, PCI_CMD_REG
2448                              <1>         ;call      pciRegRead8                  ; read PCI command register
2449 000141A2 E897F8FFFF          <1>         call       pciRegRead16
2450 000141A7 80CA05              <1>     or      dl, IO_ENA+BM_ENA               ; enable IO and bus master
2451                              <1>         ;call      pciRegWrite8
2452 000141AA E8FAF8FFFF          <1>     call    pciRegWrite16
2453                              <1>
2454                              <1>     ; 'CODEC.ASM'
2455                              <1>
2456                              <1>     ; enable codec, unmute stuff, set output rate
2457                              <1> ;   ; entry: [audio_freq] = desired sample rate
2458                              <1>
2459                              <1> ;   mov     dx, [NAMBAR]
2460                              <1> ;   add     dx, CODEC_EXT_AUDIO_CTRL_REG        ; 2Ah
2461                              <1> ;   in      ax, dx
2462                              <1> ;   or      ax, 1
2463                              <1> ;   out     dx, ax                          ; Enable variable rate audio
2464                              <1>
2465                              <1> ;       ;call      delay1_4ms
2466                              <1> ;       ;call      delay1_4ms
2467                              <1> ;       ;call      delay1_4ms
2468                              <1> ;       ;call      delay1_4ms
2469                              <1>
2470                              <1> ;   mov     ax, [audio_freq]            ; sample rate
2471                              <1>
2472                              <1> ;   mov     dx, [NAMBAR]
2473                              <1> ;   add     dx, CODEC_PCM_FRONT_DACRATE_REG      ; 2Ch
2474                              <1> ;   out     dx, ax                          ; out sample rate
2475                              <1>
2476                              <1> ;       ;call      delay1_4ms
2477                              <1> ;       ;call      delay1_4ms
2478                              <1> ;       ;call      delay1_4ms
2479                              <1> ;       ;call      delay1_4ms
2480                              <1>
2481                              <1>     ;mov    dx, [NAMBAR]                    ; mixer base address
2482                              <1>         ;add       dx, CODEC_RESET_REG          ; reset register
2483                              <1>         ;mov       ax, 42
2484                              <1>     ;out    dx, ax                           ; reset
2485                              <1>
2486                              <1>     ;mov    dx, [NABMBAR]                   ; bus master base address
2487                              <1>         ;add       dx, GLOB_STS_REG
2488                              <1>         ;mov       ax, 2
2489                              <1>     ;out    dx, ax
2490                              <1>
2491                              <1>     ; 01/06/2024
2492                              <1>     ; 16/05/2024
2493                              <1>     ; 02/12/2023
2494                              <1>     ;call   delay_100ms ; 29/05/2017
2495                              <1>
2496                              <1>     ; 02/12/2023
2497                              <1>     ;call   delay1_4ms
2498                              <1>     ;call   delay1_4ms
2499                              <1>     ;call   delay1_4ms
2500                              <1>     ;call   delay1_4ms
```

```
                                <1>
                                <1> init_ac97_codec:
                                <1>     ; 26/11/2023
                                <1>     ; 19/11/2023
                                <1>     ;   (playwav3.com, ac97_vra.asm, Erdogan Tan, 19/11/2023)
                                <1>     ; 10/06/2017
                                <1>     ; 29/05/2017
                                <1>     ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
                                <1>
                                <1>     ;; 23/11/2023 - temporary
                                <1>     ;push   ebx
                                <1>     ;mov    ebx, 0B8000h
                                <1>     ;mov    al, '?'
                                <1>     ;mov    ah, 4Eh
                                <1>     ;mov    [ebx], ax
                                <1>     ;pop    ebx
                                <1>
                                <1>     ; 01/06/2024
000141AF C605[F48C0100]01       <1>     mov     byte [reset], 1
                                <1>
                                <1>     ; 19/11/2023
000141B6 BD28000000             <1>     mov     ebp, 40 ; 21/11/2023
                                <1> _initc_1:
                                <1>     ; 26/05/2024
000141BB 66BA3000               <1>     mov     dx, GLOB_STS_REG ; 30h
000141BF 660315[B6880100]       <1>     add     dx, [NABMBAR]
000141C6 ED                     <1>     in      eax, dx
                                <1>
                                <1>     ; 01/06/2024
                                <1>     ; 02/12/2023
                                <1>     ;call   delay1_4ms
                                <1>
000141C7 83F8FF                 <1>     cmp     eax, 0FFFFFFFFh ; -1
000141CA 750A                   <1>     jne     short _initc_3
                                <1> _initc_2:
000141CC 4D                     <1>     dec     ebp     ; 21/11/2023
000141CD 7425                   <1>     jz      short _ac97_codec_ready
                                <1>
000141CF E86DF9FFFF             <1>     call    delay_100ms
000141D4 EBE5                   <1>     jmp     short _initc_1
                                <1> _initc_3:
000141D6 A900030010             <1>     test    eax, CTRL_ST_CREADY
000141DB 7517                   <1>     jnz     short _ac97_codec_ready
                                <1>
                                <1>     ; 01/06/2024
000141DD 803D[F48C0100]00       <1>     cmp     byte [reset], 0
000141E4 76E6                   <1>     jna     short _initc_2
                                <1>
000141E6 E849030000             <1>     call    reset_ac97_codec
                                <1>     ;jc     short _initc_2
                                <1>     ; 01/06/2024
000141EB C605[F48C0100]00       <1>     mov     byte [reset], 0
                                <1>     ; 26/11/2023
000141F2 EBD8                   <1>     jmp     short _initc_2
                                <1>
                                <1> _ac97_codec_ready:
000141F4 668B15[B4880100]       <1>     mov     dx, [NAMBAR]
                                <1>     ;add    dx, 0 ; ac_reg_0 ; reset register
000141FB 66EF                   <1>     out     dx, ax
                                <1>
                                <1>     ; 06/06/2024
                                <1>     ; 01/06/2024
                                <1>     ;;;
                                <1>     ;call   delay1_4ms
                                <1>     ;call   delay1_4ms
                                <1>     ;call   delay1_4ms
                                <1>     ;call   delay1_4ms
                                <1>     ;;;
                                <1>
                                <1>     ; 01/06/2024
                                <1>     ; 26/05/2024
                                <1>     ; 19/11/2023
                                <1>     ;call   delay_100ms
                                <1>     ; 24/11/2023 - temporary
                                <1>     ;call   delay_100ms
                                <1>     ;call   delay_100ms
                                <1>     ;call   delay_100ms
                                <1>
                                <1>     ; 01/06/2024
                                <1>     ;;;
000141FD 09ED                   <1>     or      ebp, ebp  ; 21/11/2023
000141FF 753E                   <1>     jnz     short _ac97_codec_init_ok
                                <1>     ;;;
                                <1>
00014201 31C0                   <1>     xor     eax, eax ; 0
00014203 668B15[B4880100]       <1>     mov     dx, [NAMBAR]
0001420A 6683C226               <1>     add     dx, CODEC_REG_POWERDOWN
0001420E 66EF                   <1>     out     dx, ax
                                <1>
                                <1>     ; 10/06/2017
                                <1>     ; 29/05/2017
                                <1>     ; wait for 1 second
                                <1>     ; 01/06/2024
                                <1>     ; 16/05/2024
00014210 B9E8030000             <1>     mov     ecx, 1000 ; 1000*0.25ms
                                <1>     ; 20/11/2023
                                <1>     ;mov    ecx, 10
                                <1>     ; 23/05/2024
                                <1>     ;mov    cl, 10
                                <1>     ; 24/11/2023 - temporary
                                <1>     ;mov    cl, 40
                                <1>     ; ecx = 40 ; 23/05/2024
                                <1> _ac97_codec_rloop:
                                <1>     ; 01/06/2024
00014215 E834F9FFFF             <1>     call    delay1_4ms
                                <1>     ; 06/06/2024
0001421A E82FF9FFFF             <1>     call    delay1_4ms
0001421F E82AF9FFFF             <1>     call    delay1_4ms
00014224 E825F9FFFF             <1>     call    delay1_4ms
                                <1>     ; 01/06/2024
                                <1>     ; 19/11/2023
                                <1>     ; 22/11/2023
                                <1>     ;push   ecx
                                <1>     ;call   delay_100ms
                                <1>     ;pop    ecx
                                <1>
                                <1>     ; 01/06/2024
                                <1>     ;;;
00014229 668B15[B4880100]       <1>     mov     dx, [NAMBAR]
00014230 6683C226               <1>     add     dx, CODEC_REG_POWERDOWN
                                <1>     ;;;
00014234 66ED                   <1>     in      ax, dx
                                <1>
                                <1>     ; 06/06/2024
```

```
                            <1>     ;call  delay1_4ms
                            <1>
                            <1>     ;and   ax, 0Fh
                            <1>     ; 21/11/2023
00014236 240F               <1>     and    al, 0Fh
00014238 3C0F               <1>     cmp    al, 0Fh
0001423A 7403               <1>     je     short _ac97_codec_init_ok
                            <1>     ; 24/11/2023 - temporary
                            <1>     ;je    short _ac97_codec_init_ok_
                            <1> _ac97_codec_yloop:
0001423C E2D7               <1>     loop   _ac97_codec_rloop
                            <1>     ; 22/11/2023
                            <1>     ; cf = 1
                            <1> init_ac97_codec_err1:
                            <1>     ;stc
                            <1> init_ac97_codec_err2:
0001423E C3                 <1>     retn
                            <1>
                            <1> ;_ac97_codec_init_ok_:
                            <1>     ;mov   dx, GLOB_STS_REG ; 30h
                            <1>     ;add   dx, [NABMBAR]
                            <1>     ;in    eax, dx
                            <1>     ;test  eax, CTRL_ST_CREADY
                            <1>     ;jnz   short _ac97_codec_init_ok
                            <1>     ;stc
                            <1>     ;retn
                            <1>
                            <1> _ac97_codec_init_ok:
                            <1>     ; 06/06/2024 (temporary)
                            <1>     ; (this may not be needed)
                            <1>     ;;;
                            <1>     ; 24/05/2024
                            <1>     ; 26/11/2023
                            <1>     ; 23/11/2023
0001423F 31C0               <1>     xor    eax, eax
                            <1>     ; 21/11/2023 - temporary
                            <1>     ; 19/11/2023
00014241 B002               <1>     mov    al, 2 ; force set 16-bit 2-channel PCM
00014243 66BA2C00           <1>     mov    dx, GLOB_CNT_REG ; 2Ch
00014247 660315[B6880100]   <1>     add    dx, [NABMBAR]
0001424E EF                 <1>     out    dx, eax
                            <1>     ;
0001424F E8FAF8FFFF         <1>     call   delay1_4ms
                            <1>     ;;;
                            <1>
                            <1>     ; 23/11/2023 - temporary
                            <1>     ;call  delay_100ms
                            <1>     ; 06/06/2024
                            <1>     ;call  delay1_4ms
                            <1>
                            <1>     ; 10/06/2017
00014254 E884020000         <1>     call   reset_ac97_controller
                            <1>
                            <1>     ; 21/11/2023
                            <1>     ;call  delay1_4ms
                            <1>
                            <1>     ; 01/06/2024
                            <1>     ; 24/05/2024
                            <1>     ; 23/05/2024
                            <1>     ; 21/11/2023 - temporary
00014259 E8E3F8FFFF         <1>     call   delay_100ms ; 06/06/2024
                            <1>
                            <1> ;   call   setup_ac97_codec
                            <1> ;
                            <1> ;detect_ac97_codec:
                            <1> ;   retn
                            <1>
                            <1> setup_ac97_codec:
                            <1>     ; 06/06/2024 - TRDOS 386 v2.0.8
                            <1>     ;;;
0001425E 668B15[B4880100]   <1>     mov    dx, [NAMBAR]
00014265 6683C22A           <1>     add    dx, CODEC_EXT_AUDIO_CTRL_REG ; 2Ah
00014269 66ED               <1>     in     ax, dx
                            <1>
                            <1>     ;and   al, ~BIT1 ; Clear DRA
0001426B 24FC               <1>     and    al, ~(BIT1+BIT0) ; Clear DRA+VRA
                            <1>     ;or    al, BIT0  ; Set VRA
                            <1>
0001426D 668B15[B4880100]   <1>     mov    dx, [NAMBAR]
00014274 6683C22A           <1>     add    dx, CODEC_EXT_AUDIO_CTRL_REG ; 2Ah
00014278 66EF               <1>     out    dx, ax
                            <1>     ;;;
                            <1>
                            <1>     ; 24/11/2023
                            <1>     ; 19/11/2023 - TRDOS 386 v2.0.7
0001427A C605[E8880100]01   <1>     mov    byte [VRA], 1
                            <1>
                            <1>     ; 25/11/2023 - temporary
                            <1>     ;jmp   short vra_not_supported
                            <1>
                            <1>     ; 23/05/2024
                            <1>     ; 18/05/2024
                            <1>     ;call  delay1_4ms ; 06/06/2024
                            <1>     ;call  delay_100ms
                            <1>
                            <1> ; 06/06/2024
                            <1> ; 01/06/2024
                            <1> %if 0
                            <1>     ; 24/05/2024
                            <1>     ; 23/05/2024
                            <1>     mov    dx, [NAMBAR]
                            <1>     add    dx, CODEC_EXT_AUDIO_REG      ; 28h
                            <1>     in     ax, dx
                            <1>
                            <1>     ; 18/05/2024
                            <1>     ; 02/12/2023
                            <1>     call   delay1_4ms
                            <1>     ; 17/05/2024
                            <1>     ;call  delay_100ms
                            <1>     ;
                            <1>     test   al, 1 ; BIT0 ; Variable Rate Audio bit
                            <1>     jz     short vra_not_supported
                            <1>
                            <1>     mov    dx, [NAMBAR]
                            <1>     add    dx, CODEC_EXT_AUDIO_CTRL_REG ; 2Ah
                            <1>     in     ax, dx
                            <1>
                            <1>     ; 06/06/2024
                            <1>     ; 23/05/2024
                            <1>     ; 02/12/2023
                            <1>     ;call  delay1_4ms
                            <1>     ;call  delay_100ms ; 18/05/2024
                            <1>     ; 24/05/2024
                            <1>     ;call  delay1_4ms
```

```
                            <1>     ;call   delay1_4ms
                            <1>     ;call   delay1_4ms
                            <1>
                            <1>     ;and    al, ~BIT1 ; Clear DRA
                            <1>     ;;;
                            <1>     ; 01/06/2024
                            <1>     and     al, ~(BIT1+BIT0) ; Clear DRA+VRA
                            <1>     ;;;
                            <1>     ; 04/06/2024
                            <1>     ;mov    dx, [NAMBAR]
                            <1>     ;add    dx, CODEC_EXT_AUDIO_CTRL_REG ; 2Ah
                            <1>     ;;;
                            <1>     out     dx, ax
                            <1> %endif
                            <1>
                            <1>     ; 04/06/2024
00014281 E8C8F8FFFF         <1>     call    delay1_4ms
                            <1>
                            <1>     ; 01/06/2024
                            <1>     ; check VRA
00014286 668B15[B4880100]   <1>     mov     dx, [NAMBAR]
0001428D 6683C228           <1>     add     dx, CODEC_EXT_AUDIO_REG       ; 28h
00014291 66ED               <1>     in      ax, dx
                            <1>
                            <1>     ; 06/06/2024
                            <1>     ;call   delay1_4ms
                            <1>
00014293 A801               <1>     test    al, 1 ; BIT0   ; Variable Rate Audio bit
00014295 746F               <1>     jz      short vra_not_supported
                            <1>
00014297 6640               <1>     inc     ax ; 0FFh -> 0
00014299 753D               <1>     jnz     short not_alc ; skip ALC850 (and ALC655) checking
                            <1>     ; ax = -1 -> 0
                            <1>
                            <1>     ;;;;
                            <1>     ; 06/06/2024 (temporary!?)
                            <1>     ; check ALC850
                            <1>     ; (which does not support VRA but it may not be detected)
                            <1>     ;
0001429B E8AEF8FFFF         <1>     call    delay1_4ms
                            <1>     ;
000142A0 668B15[B4880100]   <1>     mov     dx, [NAMBAR]
000142A7 6683C27C           <1>     add     dx, CODEC_VENDOR_ID1 ; 7Ch
000142AB 66ED               <1>     in      ax, dx
000142AD C1E010             <1>     shl     eax, 16 ; *
                            <1>     ;cmp    ax, 414Ch ; VENDOR_ID - 'AL'
                            <1>     ;jne    short not_alc
000142B0 668B15[B4880100]   <1>     mov     dx, [NAMBAR]
000142B7 6683C27E           <1>     add     dx, CODEC_VENDOR_ID2 ; 7Eh
000142BB 66ED               <1>     in      ax, dx
                            <1>     ;inc    edx
                            <1>     ;inc    edx
                            <1>     ;in     ax, dx ; VENDOR ID2
                            <1>
000142BD 40                 <1>     inc     eax ; 0FFFFFFFFh -> 0 ; 0 -> 1
000142BE 7446               <1>     jz      short vra_not_supported
000142C0 48                 <1>     dec     eax ; 1 -> 0
000142C1 7443               <1>     jz      short vra_not_supported
                            <1>
000142C3 C1C810             <1>     ror     eax, 16
000142C6 663D4C41           <1>     cmp     ax, 414Ch ; VENDOR_ID - 'AL'
000142CA 750C               <1>     jne     short not_alc
000142CC C1C010             <1>     rol     eax, 16
                            <1>
                            <1>     ; ALC850
000142CF 80FC47             <1>     cmp     ah, 47h ; 'G'
000142D2 7504               <1>     jne     short not_alc
000142D4 3C90               <1>     cmp     al, 90h
                            <1>     ;cmp    ax, 4790h ; VENDOR ID - 'G' ; bit 8-15
                            <1>                       ; CHIP ID - 1001b ; bit 4-7
                            <1>                       ; Version Number - 0 ; bit 0-3
                            <1>     ;jne    short not_alc850 ; not_alc
000142D6 742E               <1>     je      short vra_not_supported
                            <1>     ;
                            <1> ; 06/06/2024
                            <1> %if 0
                            <1> not_alc850:
                            <1>     ; ALC655
                            <1>     cmp     al, 60h
                            <1>     ;cmp    ax, 4760h ; VENDOR ID - 'G' ; bit 8-15
                            <1>                       ; CHIP ID - 0110b ; bit 4-7
                            <1>                       ; Version Number - 0 ; bit 0-3
                            <1>     ;jne    short not_alc655
                            <1>     je      short vra_not_supported
                            <1>     ;
                            <1> not_alc655:
                            <1> %endif
                            <1> not_alc:
                            <1>     ; 06/06/2024
                            <1>
000142D8 668B15[B4880100]   <1>     mov     dx, [NAMBAR]
000142DF 6683C22A           <1>     add     dx, CODEC_EXT_AUDIO_CTRL_REG         ; 2Ah
000142E3 66ED               <1>     in      ax, dx
                            <1>
                            <1>     ;and    al, ~BIT1 ; Clear DRA
                            <1>
000142E5 0C01               <1>     or      al, AC97_EA_VRA ; 1 ; 04/11/2023
                            <1>
                            <1>     ; 06/06/2024
                            <1>     ; 04/06/2024
                            <1>     ;mov    dx, CODEC_EXT_AUDIO_CTRL_REG         ; 2Ah
                            <1>     ;add    dx, [NAMBAR]
                            <1>
000142E7 66EF               <1>     out     dx, ax         ; Enable variable rate audio
                            <1>
                            <1>     ; 01/06/2024
000142E9 B90A000000         <1>     mov     ecx, 10
                            <1>     ; 21/11/2023
                            <1>     ;mov    cl, 10
                            <1> check_vra:
                            <1>     ; 01/06/2024
                            <1>     ;call   delay_100ms
000142EE E85BF8FFFF         <1>     call    delay1_4ms
                            <1>
                            <1>     ;;;
                            <1>     ; 04/06/2024
000142F3 668B15[B4880100]   <1>     mov     dx, [NAMBAR]
000142FA 6683C22A           <1>     add     dx, CODEC_EXT_AUDIO_CTRL_REG         ; 2Ah
                            <1>     ;;;
                            <1>
000142FE 66ED               <1>     in      ax, dx
                            <1>
                            <1>     ; 24/05/2024
                            <1>     ; 18/05/2024
```

```
                                <1>     ; 02/12/2023
                                <1>     ;call   delay1_4ms
                                <1>
                                <1>     ;test   al, AC97_EA_VRA ; 1
                                <1>     ; 04/06/2024
00014300 2401                   <1>     and     al, AC97_EA_VRA ; 1
00014302 7508                   <1>     jnz     short vra_ok
                                <1>
00014304 E2E8                   <1>     loop    check_vra
                                <1>
                                <1> vra_not_supported: ; 24/11/2023
                                <1>     ; VRA is not usable
                                <1>     ;mov    byte [VRA], 0
00014306 FE0D[E8880100]         <1>     dec     byte [VRA]
                                <1>
                                <1> ; 03/06/2024
                                <1> %if 0
                                <1>     ; 02/06/2024
                                <1>     jmp     short set_volume
                                <1>
                                <1>     ; 02/06/2024
                                <1> set_sampling_rate:
                                <1>     mov     dx, [NAMBAR]
                                <1>     add     dx, CODEC_PCM_FRONT_DACRATE_REG
                                <1>     out     dx, ax
                                <1>     retn
                                <1>
                                <1>     ; 02/06/2024
                                <1> get_sampling_rate:
                                <1>     mov     dx, CODEC_PCM_FRONT_DACRATE_REG
                                <1>     add     dx, [NAMBAR]
                                <1>     in      ax, dx
                                <1>     retn
                                <1> %endif
                                <1>
                                <1> vra_ok:
                                <1>
                                <1> ; 03/06/2024
                                <1> %if 0
                                <1>     ; 02/06/2024
                                <1>     ; a second test for verifying VRA status
                                <1>     ; (may be needed for ALC850 codec and CK804 controller)
                                <1>
                                <1>     call    get_sampling_rate
                                <1>     or      ax, ax
                                <1>     jz      short vra_not_supported
                                <1>     cmp     ax, 48000
                                <1>     jne     short set_volume
                                <1>     mov     ax, 24000
                                <1>     call    set_sampling_rate
                                <1>     call    get_sampling_rate
                                <1>     cmp     ax, 48000
                                <1>     je      short vra_not_supported
                                <1>     mov     ax, 48000
                                <1>     call    set_sampling_rate
                                <1>     ;call   get_sampling_rate
                                <1> %endif
                                <1>
                                <1>     ; 02/06/2024
                                <1> set_volume:
                                <1>     ; 24/05/2024
                                <1>     ; 18/05/2024 - TRDOS 386 v2.0.8
                                <1>     ; 20/11/2023
                                <1>     ; 19/11/2023 - TRDOS 386 v2.0.7
                                <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
                                <1>     ; 22/07/2020
                                <1>     ; 10/06/2017
                                <1>     ; 29/05/2017
                                <1>
                                <1>     ;mov    eax, 0202h
                                <1>     ; 21/11/2023
                                <1>     ;mov    eax, 0B0Bh
                                <1>     ; 23/11/2023
0001430C B81D1D0000             <1>     mov     eax, 1D1Dh
00014311 66A3[EC880100]         <1>     mov     [audio_master_volume], ax
                                <1>     ; 24/05/2024
00014317 66A3[EE880100]         <1>     mov     [audio_pcmo_volume], ax
                                <1>     ;mov    ax, 1F1Fh ; 31, 31
                                <1>
                                <1>     ; 21/11/2023
                                <1>     ;push   eax
                                <1>
                                <1>     ; 20/11/2023
                                <1>     ;mov    ax, 0B0Bh
                                <1>     ; 23/11/2023
0001431D 66B80202               <1>     mov     ax, 0202h
                                <1>
00014321 668B15[B4880100]       <1>     mov     dx, [NAMBAR]
00014328 6683C202               <1>     add     dx, CODEC_MASTER_VOL_REG      ;02h
                                <1>     ;xor    ax, ax  ; volume attenuation = 0 (max. volume)
                                <1>     ; 06/08/2022
                                <1>     ;xor    eax, eax
                                <1>     ; 19/11/2023
0001432C 66EF                   <1>     out     dx, ax
                                <1>
                                <1>     ; 01/06/2024
                                <1>     ; 23/05/2024
                                <1>     ; 16/05/2024
                                <1>     ; 20/11/2023
0001432E E81BF8FFFF             <1>     call    delay1_4ms
                                <1>     ;call   delay1_4ms
                                <1>     ;call   delay1_4ms
                                <1>     ;call   delay1_4ms
                                <1>     ; 21/11/2023 temporary
                                <1>     ;call   delay_100ms ; 18/05/2024
                                <1>
                                <1>     ; 21/11/2023
                                <1>     ;pop    eax
                                <1>
                                <1>     ; 23/11/2023
                                <1>     ; 21/11/2023
                                <1>     ;mov    ax, 0202h
                                <1>
00014333 668B15[B4880100]       <1>     mov     dx, [NAMBAR]
0001433A 6683C218               <1>     add     dx, CODEC_PCM_OUT_REG         ;18h
                                <1>     ;xor    ax, ax
0001433E 66EF                   <1>     out     dx, ax
                                <1>
                                <1>     ; 01/06/2024
                                <1>     ; 23/05/2024
                                <1>     ; 16/05/2024
                                <1>     ; 20/11/2023
                                <1>         ;call      delay1_4ms
                                <1>         ;call      delay1_4ms
```

```
                        <1>         ;call      delay1_4ms
                        <1>     ;call  delay1_4ms
                        <1>     ; 21/11/2023 - temporary
                        <1>     ;call  delay_100ms ; 18/05/2024
                        <1>
                        <1> ; 23/11/2023
                        <1> %if 0
                        <1>     mov    dx, [NAMBAR]
                        <1>     add    dx, CODEC_MASTER_MONO_VOL_REG ;06h
                        <1>     ;xor   ax, ax
                        <1>     out    dx, ax
                        <1>
                        <1>         ;call      delay1_4ms
                        <1>         ;call      delay1_4ms
                        <1>         ;call      delay1_4ms
                        <1>     ;call  delay1_4ms
                        <1>
                        <1>     mov    dx, [NAMBAR]
                        <1>     add    dx, CODEC_PCBEEP_VOL_REG     ;0Ah
                        <1>     ;xor   ax, ax
                        <1>     out    dx, ax
                        <1>
                        <1>         ;call      delay1_4ms
                        <1>         ;call      delay1_4ms
                        <1>         ;call      delay1_4ms
                        <1>     ;call  delay1_4ms
                        <1>
                        <1>     mov    ax, 8008h ; Mute
                        <1>     mov    dx, [NAMBAR]
                        <1>     ; 22/07/2020
                        <1>     add    dx, CODEC_PHONE_VOL_REG              ;0Ch
                        <1>                              ; AC97_PHONE_VOL ; TAD Input (Mono)
                        <1>     out    dx, ax
                        <1>
                        <1>         ;call      delay1_4ms
                        <1>         ;call      delay1_4ms
                        <1>         ;call      delay1_4ms
                        <1>     ;call  delay1_4ms
                        <1>
                        <1>         mov ax, 0808h
                        <1>     mov    dx, [NAMBAR]
                        <1>         add dx, CODEC_LINE_IN_VOL_REG ;10h ; Line Input (Stereo)
                        <1>     out    dx, ax
                        <1>
                        <1>         ;call      delay1_4ms
                        <1>         ;call      delay1_4ms
                        <1>         ;call      delay1_4ms
                        <1>     ;call  delay1_4ms
                        <1>
                        <1>     ;mov   ax, 0808h
                        <1>     mov    dx, [NAMBAR]
                        <1>         add dx, CODEC_CD_VOL_REG ;12h ; CR Input (Stereo)
                        <1>     out    dx, ax
                        <1>
                        <1>         ;call      delay1_4ms
                        <1>         ;call      delay1_4ms
                        <1>         ;call      delay1_4ms
                        <1>     ;call  delay1_4ms
                        <1>
                        <1>     ;mov   ax, 0808h
                        <1>     mov    dx, [NAMBAR]
                        <1>         add dx, CODEC_AUX_VOL_REG ;16h ; Aux Input (Stereo)
                        <1>     out    dx, ax
                        <1>
                        <1>         ;call      delay1_4ms
                        <1>         ;call      delay1_4ms
                        <1>         ;call      delay1_4ms
                        <1>         ;call      delay1_4ms
                        <1>
                        <1>     ; 21/11/2023 - temporary
                        <1>     ;call  delay_100ms
                        <1> %endif
                        <1>     ; 16/05/2024
                        <1>     ;clc
                        <1>
                        <1> set_volume_ok:
                        <1>
                        <1> ;detect_ac97_codec:
00014340 C3             <1>     retn
                        <1>
                        <1> set_ac97_bdl: ; Set AC97 (ICH) Buffer Descriptor List
                        <1>     ; 04/06/2024 - TRDOS 386 v2.0.8
                        <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
                        <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>
                        <1>     ; 04/06/2024
                        <1>     ; ecx = DMA half buffer size as truncated for 8 byte alignment
                        <1>
                        <1>     ;shr   ecx, 1 ; dma half buffer size
00014341 89CE           <1>     mov    esi, ecx
                        <1>
00014343 BF[F0880100]   <1>         mov     edi, audio_bdl_buff      ; get BDL address
                        <1>     ;mov   ecx, 32 / 2           ; make 32 entries in BDL
                        <1>     ; 06/08/2022
00014348 29C9           <1>     sub    ecx, ecx
0001434A B110           <1>     mov    cl, 16
                        <1>
0001434C 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>
0001434E A1[D0880100]   <1>     mov    eax, [audio_dma_buff] ; Physical address of DMA buffer
                        <1>
                        <1> s_ac97_bdl1:
00014353 AB             <1>     stosd                        ; store dmabuffer1 address
                        <1>
00014354 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> ;IOCequ     BIT31                ; Fire an interrupt whenever this
```

```
                                <1>                                     ; buffer is complete.
                                <1>
                                <1> ;BUPequ     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>
00014356 89F0                   <1>     mov     eax, esi ; DMA half buffer size
                                <1>
                                <1>             ; 04/06/2024 - Erdogan Tan
                                <1>             ; NOTE: I have changed DMA half buffer size to truncated
                                <1>             ; (8 byte aligned) audio buffer size for smooth audio playing
                                <1>             ; (it was page border aligned before June 4, 2024)
                                <1>             ;
                                <1>             ; For example:
                                <1>             ;    For 65416 bytes audio buffer (22kHz, 16bit samples)
                                <1>             ;       (user's audio buff virtual addr is mapped to physical)
                                <1>             ;      ((kernel's and user's audio buff pages are same))
                                <1>             ;    1) Memory allocation (of it) is 65536 bytes
                                <1>             ;    2) DMA half buffer size is 65416 bytes
                                <1>             ;       (it would be 65536 bytes before this modification)
                                <1>             ;      ((additional 140 bytes would cause to a noise))
                                <1>             ;    3) Total DMA buffer size is 131072 bytes
                                <1>             ;       (the last 240 bytes will not be used for playing)
                                <1>             ;
                                <1>             ; (buffer will be truncated if the size is not a multiple of 8)
                                <1>
00014358 01C2                   <1>     add     edx, eax
0001435A D1E8                   <1>     shr     eax, 1  ; count of 16 bit samples
                                <1>     ; 19/11/2023
0001435C 0D000000C0             <1>     or      eax, IOC+BUP
                                <1>     ;or     eax, IOC ; 11/06/2017
00014361 AB                     <1>     stosd
                                <1>
                                <1> ; 2nd buffer:
                                <1>
00014362 89D0                   <1>         mov eax, edx ; Physical address of the 2nd half of DMA buffer
00014364 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> ;
00014365 89F0                   <1>     mov     eax, esi ; DMA half buffer size
00014367 D1E8                   <1>     shr     eax, 1 ; count of 16 bit samples
                                <1>     ; 19/11/2023
00014369 0D000000C0             <1>     or      eax, IOC+BUP
                                <1>     ;or     eax, IOC ; 11/06/2017
0001436E AB                     <1>     stosd
                                <1>
0001436F E2DD                   <1>     loop    s_ac97_bdl0
                                <1>
00014371 C3                     <1>     retn
                                <1>
                                <1> ac97_start_play:
                                <1>     ; 26/05/2024 - TRDOS 386 v2.0.8
                                <1>     ; 26/11/2023
                                <1>     ; 20/11/2023
                                <1>     ; 19/11/2023 - TRDOS 386 v2.0.7
                                <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
                                <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>
                                <1>     ; 21/11/2023 - temporary
                                <1>     ;call   ac97_codec_config
                                <1>
                                <1> ; 20/11/2023
                                <1> %if 0
                                <1>     AC97_EA_VRA equ 0001h ; 04/11/2023
                                <1>
                                <1>     mov     dx, [NAMBAR]
                                <1>     add     dx, CODEC_EXT_AUDIO_CTRL_REG        ; 2Ah
                                <1>     in      ax, dx
                                <1>
                                <1>     ; 23/05/2024
                                <1>     call    delay1_4ms
                                <1>
                                <1>     and     al, ~BIT1 ; Clear DRA
                                <1>     or      al, AC97_EA_VRA ; 1 ; 04/11/2023
                                <1>     out     dx, ax                  ; Enable variable rate audio
                                <1>
                                <1>        ;call       delay1_4ms
                                <1>        ;call       delay1_4ms
                                <1>        ;call       delay1_4ms
                                <1>        ;call       delay1_4ms
                                <1>
                                <1>     ; 23/05/2024
                                <1>     call    delay_100ms
                                <1> %endif
                                <1>
                                <1>     ; 26/11/2023
                                <1>     ; 24/11/2023 - temporary
                                <1>     ;mov    ax, 48000
                                <1>
                                <1>     ; 20/11/2023
00014372 803D[E8880100]01       <1>     cmp     byte [VRA], 1
00014379 7213                   <1>     jb      short skip_set_rate
                                <1>
0001437B 66A1[E6880100]         <1>     mov     ax, [audio_freq]            ; sample rate
                                <1>
                                <1> ;skip_set_rate:    ; 24/11/2023
00014381 668B15[B4880100]       <1>     mov     dx, [NAMBAR]
00014388 6683C22C               <1>     add     dx, CODEC_PCM_FRONT_DACRATE_REG     ; 2Ch
0001438C 66EF                   <1>     out     dx, ax                      ; out sample rate
                                <1>
                                <1>     ; 01/06/2024
```

```
                                <1>     ; 27/11/2023
                                <1>     ;call   delay1_4ms
                                <1>     ;call   delay1_4ms
                                <1>     ;call   delay1_4ms
                                <1>     ;call   delay1_4ms
                                <1>     ; 27/11/2023 - temporary
                                <1>     ;call   delay_100ms
                                <1>
                                <1>     ; 26/11/2023
                                <1> skip_set_rate:     ; 20/11/2023
                                <1>
                                <1>     ;; 23/11/2023 - temporary
                                <1>     ;push   ebx
                                <1>     ;mov    ebx, 0B8000h
                                <1>     ;mov    al, 'x'
                                <1>     ;mov    ah, 4Eh
                                <1>     ;mov    [ebx], ax
                                <1>     ;pop    ebx
                                <1> ;
                                <1> ; register reset the DMA engine. This may cause a pop noise on the output
                                <1> ; lines when the device is reset. Prolly a better idea to mute output, then
                                <1> ; reset.
                                <1> ;
                                <1>     ;; 21/11/2023 - temporary
                                <1>     ;; 20/11/2023
                                <1>     ;mov    dx, [NABMBAR]
                                <1>     ;add    dx, PO_CR_REG               ; set pointer to Cntl reg
                                <1>     ;mov    al, RR                      ; set reset
                                <1>     ;out    dx, al                      ; self clearing bit
                                <1>
                                <1>     ; 23/11/2023 - temporary
                                <1>     ;call   delay_100ms
                                <1>
                                <1> ;   mov     edi, audio_bdl_buff
                                <1> ;   mov     edx, [audio_dmabuff_size]
                                <1> ;   shr     edx, 1
                                <1> ;   mov     ecx, 32/2
                                <1> ;ac97_set_bdl_buffer:
                                <1> ;   ; 1st half of DMA buffer
                                <1> ;   mov     eax, [audio_dma_buff]
                                <1> ;   push    eax
                                <1> ;   stosd
                                <1> ;   mov     eax, edx ; dma buffer size / 2
                                <1> ;   or      eax, IOC+BUP
                                <1> ;   stosd
                                <1> ;   pop     eax
                                <1> ;   ; 2nd half of DMA buffer
                                <1> ;   add     eax, edx
                                <1> ;   stosd
                                <1> ;   mov     eax, edx ; dma buffer size / 2
                                <1> ;   or      eax, IOC+BUP
                                <1> ;   stosd
                                <1> ;   loop    ac97_set_bdl_buffer
                                <1>
                                <1> ; tell the DMA engine where to find our list of Buffer Descriptors.
                                <1> ; this 32bit value is a flat mode memory offset (ie no segment:offset)
                                <1> ;
                                <1> ; write NABMBAR+10h with offset of buffer descriptor list
                                <1> ;
0001438E B8[F0880100]           <1>         mov eax, audio_bdl_buff
00014393 668B15[B6880100]       <1>     mov     dx, [NABMBAR]
0001439A 6683C210               <1>     add     dx, PO_BDBAR_REG
0001439E EF                     <1>     out     dx, eax
                                <1>
                                <1>     ; 01/06/2024
                                <1>     ; 24/05/2024
                                <1>     ; 23/11/2023 - temporary
                                <1>     ;call   delay_100ms
                                <1>     ; 23/05/2024
                                <1>     ;call   delay1_4ms
                                <1> ;
                                <1> ; All set. Let's play some music.
                                <1> ;
                                <1>     ;mov    eax, 31
                                <1>     ; 06/08/2022
0001439F 29C0                   <1>     sub     eax, eax
000143A1 B01F                   <1>     mov     al, 31
000143A3 E846000000             <1>     call    set_ac97_LastValidIndex
                                <1>
                                <1>     ; 01/06/2024
                                <1>     ; 24/05/2024
                                <1>     ; 23/11/2023 - temporary
                                <1>     ;call   delay_100ms
                                <1>     ; 23/05/2024
                                <1>     ;call   delay1_4ms
                                <1>     ; 27/05/2024
                                <1>     ;call   delay1_4ms
                                <1>     ;call   delay1_4ms
                                <1>     ;call   delay1_4ms
                                <1>
000143A8 C605[EA880100]01       <1>     mov     byte [audio_play_cmd], 1 ; play command (do not stop) !
                                <1>
                                <1> ac97_play: ; continue to play (after pause)
                                <1>     ; 27/05/2024
                                <1>     ;;;
                                <1>     ; 19/05/2024 - temporary
                                <1>     ; 24/11/2023 - temporary
                                <1>     ;mov    ax, 0202h
                                <1>     ; 23/05/2024
000143AF 66B81F1F               <1>     mov     ax, 1F1Fh
000143B3 662B05[EC880100]       <1>     sub     ax, [audio_master_volume]
                                <1>
000143BA 668B15[B4880100]       <1>     mov     dx, [NAMBAR]
000143C1 6683C202               <1>     add     dx, CODEC_MASTER_VOL_REG  ; 02h ; Line Out
000143C5 66EF                   <1>     out     dx, ax
                                <1>
                                <1>     ; 01/06/2024
                                <1>     ; 19/05/2024 - temporary
                                <1>     ;call   delay_100ms
                                <1>     ; 23/05/2024
                                <1>     ;call   delay1_4ms
                                <1>     ; 24/05/2024
                                <1>     ;call   delay1_4ms
                                <1>     ;call   delay1_4ms
                                <1>     ;call   delay1_4ms
                                <1>
                                <1>     ;mov    ax, 0202h
                                <1>     ; 23/05/2024
                                <1>     ;mov    ax, 1F1Fh
                                <1>     ;sub    ax, [audio_master_volume]
                                <1>
                                <1>     ; 27/05/2024
000143C7 66B81F1F               <1>     mov     ax, 1F1Fh
000143CB 662B05[EE880100]       <1>     sub     ax, [audio_pcmo_volume]
```

```
                                <1>
000143D2 668B15[B4880100]       <1>     mov     dx, [NAMBAR]
000143D9 6683C218               <1>     add     dx, CODEC_PCM_OUT_REG ; 18h ; PCM Out
000143DD 66EF                   <1>     out     dx, ax
                                <1>
                                <1>     ; 01/06/2024
                                <1>     ; 24/05/2024
                                <1>     ; 23/05/2024
                                <1>     ;call   delay1_4ms
                                <1>     ;call   delay1_4ms
                                <1>     ;call   delay1_4ms
                                <1>     ;call   delay1_4ms
                                <1>     ; 24/05/2024
                                <1>     ;call   delay_100ms
                                <1>     ;;;
                                <1>
                                <1>     ; 11/06/2017
                                <1>     ; 29/05/2017
                                <1>     ; 28/05/2017
000143DF 668B15[B6880100]       <1>         mov     dx, [NABMBAR]
000143E6 6683C21B               <1>         add     dx, PO_CR_REG           ; PCM out control register
                                <1>     ; 26/11/2023
000143EA B011                   <1>     mov     al, IOCE+RPBM ; 29/05/2017
                                <1>     ; 24/11/2023
                                <1>     ;mov    al, 1Dh ; (Ref: KolibriOS, intelac97.asm, 'play:')
000143EC EE                     <1>     out      dx, al                 ; set start!
                                <1>
                                <1>     ;mov    byte [audio_play_cmd], 1 ; play command (do not stop) !
                                <1>
000143ED C3                     <1>     retn
                                <1>
                                <1> ;input AL = index # to stop on
                                <1> set_ac97_LastValidIndex:
                                <1>     ; 28/05/2017
                                <1>     ; Derived from 'setLastValidIndex' procedure in 'ICHWAV.ASM'
                                <1>     ; .wav player for DOS by Jeff Leyda (02/09/2002)
000143EE 668B15[B6880100]       <1>     mov     dx, [NABMBAR]
000143F5 6683C215               <1>     add     dx, PO_LVI_REG
000143F9 EE                     <1>         out     dx, al
                                <1>     ;mov    [audio_lvi], al ; for ac97_int_handler
000143FA C3                     <1>     retn
                                <1>
                                <1> ac97_volume:
                                <1>     ; 23/05/2024
                                <1>     ; 28/05/2017
                                <1>     ; bl = component (0 = master/playback/lineout volume)
                                <1>     ;                (1 = PCM Out volume) ; 23/05/2024
                                <1>     ; cl = left channel volume level (0 to 31)
                                <1>     ; ch = right channel volume level (0 to 31)
                                <1>
                                <1>     ; 24/05/2024
000143FB 80FB01                 <1>     cmp     bl, 1
000143FE 7727                   <1>     ja      short ac97_vol_2 ; temporary !
                                <1>
00014400 66B81F1F               <1>     mov     ax, 1F1Fh ; 31,31
00014404 38C1                   <1>     cmp     cl, al
00014406 771F                   <1>     ja      short ac97_vol_2
00014408 38E5                   <1>     cmp     ch, ah
0001440A 771B                   <1>     ja      short ac97_vol_2
                                <1>
0001440C 08DB                   <1>     or      bl, bl
0001440E 7518                   <1>     jnz     short ac97_vol_1
                                <1>
                                <1>     ; 23/05/2024
00014410 66BA0200               <1>     mov     dx, CODEC_MASTER_VOL_REG  ; 02h ; Line Out
                                <1>     ; 24/05/2024
00014414 66890D[EC880100]       <1>     mov     [audio_master_volume], cx
                                <1> ac97_vol_0:
0001441B 6629C8                 <1>     sub     ax, cx
                                <1>     ;mov    dx, [NAMBAR]
                                <1>     ;add    dx, CODEC_MASTER_VOL_REG
                                <1>     ; 23/05/2024
0001441E 660315[B4880100]       <1>     add     dx, [NAMBAR]
00014425 66EF                   <1>     out      dx, ax
                                <1>     ; 23/05/2024
                                <1> ac97_vol_2:
                                <1>     ; 21/11/2023
                                <1> ;_ac97_ih5: ; 06/08/2022
00014427 C3                     <1>     retn
                                <1>
                                <1> ac97_vol_1:
                                <1>     ; 24/05/2024
                                <1>     ; 23/05/2024
                                <1>     ;cmp    bl, 1
                                <1>     ;ja     short ac97_vol_2 ; temporary !
                                <1>     ; BL = 1
                                <1>     ; PCM OUT Volume
00014428 66BA1800               <1>     mov     dx, CODEC_PCM_OUT_REG ; 18h ; PCM out
                                <1>     ; 24/05/2024
0001442C 66890D[EE880100]       <1>     mov     [audio_pcmo_volume], cx
00014433 EBE6                   <1>     jmp     short ac97_vol_0
                                <1>
                                <1> ac97_int_handler:
                                <1>     ; 04/06/2024
                                <1>     ; 03/06/2024
                                <1>     ; 02/06/2024 - TRDOS 386 v2.0.8
                                <1>     ; 27/11/2023
                                <1>     ; 24/11/2023
                                <1>     ; 21/11/2023
                                <1>     ; 20/11/2023 - TRDOS 386 v2.0.7
                                <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
                                <1>     ; 12/10/2017
                                <1>     ; 10/10/2017
                                <1>     ; 09/10/2017
                                <1>     ; 13/06/2017, 13/06/2017
                                <1>     ; 10/06/2017, 11/06/2017
                                <1>     ; Interrupt Handler for AC97 (ICH) Audio Controller
                                <1>     ; Note: called by 'dev_IRQ_service'
                                <1>     ; 28/05/2017
                                <1>
                                <1>     ;push   eax ; * must be saved !
                                <1>     ;push   edx
                                <1>     ;push   ecx
                                <1>     ;push   ebx ; * must be saved !
                                <1>     ;push   esi
                                <1>     ;push   edi
                                <1>
                                <1> ; 02/06/2024
                                <1> %if 0
                                <1>
                                <1>     ;cmp    byte [audio_busy], 1
                                <1>     ;jnb    _ac97_ih2 ; busy !
                                <1>
                                <1>     ; 24/11/2023
```

```
                                <1>         ;mov       dx, [NABMBAR]
                                <1>     ;mov    dx, PO_SR_REG
                                <1>     ;in     ax, dx
                                <1>     ;test   al, BCIS ; bit 3, 8
                                <1>     ;jz     short _ac97_ih5
                                <1>
                                <1>     ; 24/11/2023
                                <1>     mov     dx, GLOB_STS_REG
                                <1>         add dx, [NABMBAR]
                                <1>     in      eax, dx
                                <1>
                                <1>     cmp     eax, 0FFFFFFFFh ; -1
                                <1>     ;je     _ac97_ih3 ; exit
                                <1>     ; 06/08/2022
                                <1>     je      short _ac97_ih5
                                <1>
                                <1>     ;test   eax, 40h ; PCM Out Interrupt
                                <1>     ; 06/08/2022
                                <1>     test    al, 40h
                                <1>     jnz     short _ac97_ih0
                                <1>
                                <1>     ; 24/11/2023
                                <1>     test    eax, eax
                                <1>     ;jz     _ac97_ih3 ; exit
                                <1>     ; 06/08/2022
                                <1>     jz      short _ac97_ih5
                                <1>
                                <1>     ;mov    dx, GLOB_STS_REG
                                <1>         ;add       dx, [NABMBAR]
                                <1>     out     dx, eax
                                <1>
                                <1>     ;jmp    _ac97_ih3 ; exit
                                <1>
                                <1>     ; 06/08/2022
                                <1>     retn
                                <1>     ; 21/11/2023
                                <1>     ;jmp    short _ac97_ih5
                                <1>
                                <1> _ac97_ih0:
                                <1>     ; 24/11/2023
                                <1>     ;push   eax
                                <1>     ; 09/10/2017
                                <1>     cmp     byte [audio_play_cmd], 1
                                <1>     ;jb     short _ac97_ih4 ; stop command !
                                <1>     ; 24/11/2023
                                <1>     jnb     short _ac97_ih4
                                <1>
                                <1>     ;mov    dx, GLOB_STS_REG
                                <1>         ;add       dx, [NABMBAR]
                                <1>     out     dx, eax
                                <1>     ;jmp    short _ac97_stop
                                <1>
                                <1>     ; 24/11/2023
                                <1> ac97_stop:
                                <1>     ; 28/05/2017
                                <1>     mov     byte [audio_play_cmd], 0 ; stop !
                                <1>
                                <1> %else
                                <1>     ; 02/06/2024
                                <1> _ac97_ih0:
00014435 803D[EA880100]01       <1>     cmp     byte [audio_play_cmd], 1
0001443C 7334                   <1>     jnb     short _ac97_ih4
                                <1> %endif
                                <1>
                                <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
0001443E 30C0                   <1>     xor     al, al ; 0
00014440 E813000000             <1>     call    ac97_po_cmd
                                <1>
                                <1>     ; (Ref: KolibriOS, intelac97.asm, 'stop:')
                                <1>     ; Clear FIFOE, BCIS, LVBCI (Ref: Intel ICH hub manual)
00014445 66B81C00               <1>     mov     ax, 1Ch
00014449 668B15[B6880100]       <1>     mov     dx, [NABMBAR]
00014450 6683C216               <1>     add     dx, PO_SR_REG
00014454 66EF                   <1>     out     dx, ax
                                <1>
                                <1>     ;retn
                                <1>
                                <1>     ; 11/06/2017
00014456 B002                   <1>     mov     al, RR
                                <1> ac97_po_cmd:
                                <1>      ;11/06/2017
                                <1>     ; 29/05/2017
00014458 668B15[B6880100]       <1>     mov     dx, [NABMBAR]
0001445F 6683C21B               <1>         add     dx, PO_CR_REG           ; PCM out control register
00014463 EE                     <1>     out     dx, al
00014464 C3                     <1>     retn
                                <1>
                                <1> ; 02/06/2024
                                <1> %if 1
                                <1>     ; 24/11/2023
                                <1> ac97_stop:
                                <1>     ; 28/05/2017
00014465 C605[EA880100]00       <1>     mov     byte [audio_play_cmd], 0 ; stop !
0001446C EBD0                   <1>     jmp     short _ac97_stop
                                <1> %endif
                                <1>
                                <1> ac97_pause:
                                <1>     ; 11/06/2017
                                <1>     ; 29/05/2017
0001446E B010                   <1>     mov     al, IOCE
00014470 EBE6                   <1>     jmp     short ac97_po_cmd
                                <1>
                                <1> _ac97_ih4:
                                <1>     ;mov    byte [audio_busy], 1
                                <1>
                                <1> ; 02/06/2024
                                <1> %if 1
                                <1>     ; 24/11/2023 (TRDOS386 'audio.s')
                                <1>         mov dx, [NABMBAR]
00014472 668B15[B6880100]       <1>     add     dx, PO_SR_REG
00014479 6683C216               <1>     in      ax, dx
0001447D 66ED                   <1>
                                <1>     test    al, BCIS ; bit 3, 8
0001447F A808                   <1>     jz      short _ac97_ih2         ; 02/06/2024
00014481 744C                   <1>
                                <1>     ; 02/06/2024
00014483 50                     <1>     push    eax ; *
```

```
                                <1> %else
                                <1>     ; 27/05/2024
                                <1>     ; 24/11/2023
                                <1>     push    eax
                                <1>
                                <1>     mov     ax, 1Ch ; FIFOE(=16)+BCIS(=8)+LVBCI(=4)
                                <1>     mov     dx, PO_SR_REG
                                <1>         add dx, [NABMBAR]
                                <1>     out     dx, ax
                                <1>
                                <1>     mov     dx, PO_CIV_REG
                                <1>         add dx, [NABMBAR]
                                <1>     in      al, dx
                                <1>
                                <1>     ;cmp    al, [audio_civ] ; [audio_flag]
                                <1>     ;je     short _ac97_ih2
                                <1>
                                <1>     ; 21/11/2023
                                <1>     ;mov    [audio_civ], al
                                <1>     ; 20/11/2023
                                <1>     mov     ah, al
                                <1>     dec     al
                                <1>     ;inc    al ; 11/06/2017
                                <1>     and     al, 1Fh
                                <1> %endif
                                <1>     ; 02/06/2024
00014484 66BA1400               <1>     mov     dx, PO_CIV_REG
00014488 660315[B6880100]       <1>         add dx, [NABMBAR]
                                <1>     ;in     ax, dx
                                <1>     ; 03/06/2024
0001448F EC                     <1>     in      al, dx
                                <1>                     ; al = CVI, ah = LVI
                                <1>     ;xchg   ah, al
                                <1>     ;               ; al = LVI, ah = CVI
                                <1>     ;cmp    al, ah
                                <1>     ;jne    short _ac97_ih5
                                <1>     ; 03/06/2024
00014490 88C4                   <1>     mov     ah, al
                                <1>
00014492 FEC8                   <1>     dec     al
00014494 241F                   <1>     and     al, 1Fh
                                <1>
                                <1>     ;mov    dx, PO_LVI_REG
                                <1>     ;add    dx, [NABMBAR]
00014496 6642                   <1>         inc dx      ; 02/06/2024
00014498 EE                     <1>     out     dx, al
                                <1>
                                <1>     ; 12/10/2017
                                <1>     ;mov    al, [audio_civ]
                                <1>     ;inc    al
                                <1>     ;and    al, 1
                                <1>     ;mov    [audio_flag], al
                                <1>     ; 27/11/2023
                                <1>     ; 20/11/2023
                                <1>     ;inc    ah
                                <1> _ac97_ih5:  ; 02/06/2024
00014499 80E401                 <1>     and     ah, 1
0001449C 8825[DC880100]         <1>     mov     [audio_flag], ah
                                <1>
                                <1>     ;; [audio_flag] : 0 = Buffer 1, 1 = Buffer 2
                                <1>     ;
                                <1>     ; 21/11/2023
                                <1>     ;call   ac97_tuneloop
                                <1>
                                <1>     ; 24/11/2023
                                <1> ac97_tuneloop:
                                <1>     ; 09/10/2017
000144A2 8B3D[D0880100]         <1>     mov     edi, [audio_dma_buff]
                                <1>     ; 04/06/2024
                                <1>     ;mov    ecx, [audio_dmabuff_size]
                                <1>     ;shr    ecx, 1 ; dma buff size / 2 = half buffer size
000144A8 8B0D[CC880100]         <1>     mov     ecx, [audio_buff_size]
000144AE 80E1F8                 <1>     and     cl, ~7 ; 8 byte aligned
                                <1>
                                <1>     ; 12/10/2017
000144B1 803D[DC880100]00       <1>     cmp     byte [audio_flag], 0
000144B8 7702                   <1>     ja      short _ac97_ih1  ; Playing Half Buffer 2 (Current: FLAG)
                                <1>     ; Playing Half Buffer 1 (Current: EOL)
000144BA 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>
000144BC 8B35[C8880100]         <1>     mov     esi, [audio_p_buffer] ; phy addr of audio buff
000144C2 C1E902                 <1>     shr     ecx, 2 ; half buff size / 4
000144C5 F3A5                   <1>     rep     movsd
                                <1>
                                <1>     ; 10/10/2017
                                <1>     ; switch flag value
000144C7 8035[DC880100]01       <1>     xor     byte [audio_flag], 1
                                <1>
                                <1> ; 02/06/2024
                                <1> %if 1
                                <1>     ; 02/06/2024
000144CE 58                     <1>     pop     eax ; *
                                <1> _ac97_ih2:
                                <1>     ;mov    ax, 1Ch ; FIFOE(=16)+BCIS(=8)+LVBCI(=4)
000144CF 66BA1600               <1>     mov     dx, PO_SR_REG
000144D3 660315[B6880100]       <1>     add     dx, [NABMBAR]
000144DA 66EF                   <1>     out     dx, ax
                                <1> %else
                                <1>     ; 27/11/2023
                                <1>     ; 21/11/2023 - temporary
                                <1>     ;push   ebx
                                <1>     ;mov    ebx, 0B8002h
                                <1>     ;mov    al, [audio_flag]
                                <1>     ;add    al, '1'
                                <1>     ;mov    ah, 0Fh
                                <1>     ;mov    [ebx], ax
                                <1>     ;pop    ebx
                                <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>     ; 24/11/2023
                                <1>     ;retn
                                <1>
                                <1>     ;
                                <1>     ; 24/11/2023
                                <1>     pop     eax
                                <1>     ; 24/11/2023
                                <1>     ; 20/11/2023
```

```
                            <1>     ;and    eax, 40h
                            <1>         mov dx, [NABMBAR]
                            <1>     add     dx, GLOB_STS_REG
                            <1>     out     dx, eax
                            <1>
                            <1>     ;; 13/06/2017
                            <1>     ;mov    al, 11h ; IOCE + RPBM
                            <1>     ;mov    dx, PO_CR_REG
                            <1>         ;add        dx, [NABMBAR]
                            <1>     ;out    dx, al
                            <1>
                            <1>     ; 24/11/2023
                            <1>     ;mov    dx, [NABMBAR]
                            <1>     ;add    dx, PO_SR_REG  ; set pointer to Status reg
                            <1>     ;mov    ax, 1Ch
                            <1>     ;out    dx, ax
                            <1>
                            <1>     ; 24/11/2023
                            <1> _ac97_ih2:
                            <1>     ;mov    byte [audio_busy], 0
                            <1> %endif
                            <1>
                            <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>
000144DC C3                 <1>     retn
                            <1>
                            <1> reset_ac97_controller:
                            <1>     ; 06/06/2024
                            <1>     ; 01/06/2024
                            <1>     ; 27/05/2024
                            <1>     ; 16/05/2024
                            <1>     ; 10/06/2017
                            <1>     ; 29/05/2017
                            <1>     ; 28/05/2017
                            <1>     ; reset AC97 audio controller registers
000144DD 31C0               <1>     xor     eax, eax
000144DF 66BA0B00           <1>         mov dx, PI_CR_REG
000144E3 660315[B6880100]   <1>     add     dx, [NABMBAR]
000144EA EE                 <1>     out     dx, al
                            <1>
                            <1>     ; 16/05/2024
                            <1>     ;call   delay1_4ms
                            <1>
000144EB 66BA1B00           <1>         mov     dx, PO_CR_REG
000144EF 660315[B6880100]   <1>     add     dx, [NABMBAR]
000144F6 EE                 <1>     out     dx, al
                            <1>
                            <1>     ; 16/05/2024
                            <1>     ;call   delay1_4ms
                            <1>
000144F7 66BA2B00           <1>         mov     dx, MC_CR_REG
000144FB 660315[B6880100]   <1>     add     dx, [NABMBAR]
00014502 EE                 <1>     out     dx, al
                            <1>
                            <1>     ; 16/05/2024
                            <1>     ;call   delay1_4ms
                            <1>
00014503 B002               <1>         mov     al, RR
00014505 66BA0B00           <1>         mov     dx, PI_CR_REG
00014509 660315[B6880100]   <1>     add     dx, [NABMBAR]
00014510 EE                 <1>     out     dx, al
                            <1>
                            <1>     ; 16/05/2024
                            <1>     ;call   delay1_4ms
                            <1>
00014511 66BA1B00           <1>         mov     dx, PO_CR_REG
00014515 660315[B6880100]   <1>     add     dx, [NABMBAR]
0001451C EE                 <1>     out     dx, al
                            <1>
                            <1>     ; 16/05/2024
                            <1>     ;call   delay1_4ms
                            <1>
0001451D 66BA2B00           <1>         mov     dx, MC_CR_REG
00014521 660315[B6880100]   <1>     add     dx, [NABMBAR]
00014528 EE                 <1>     out     dx, al
                            <1>
                            <1>     ; 27/05/2024
                            <1>     ; 16/05/2024
                            <1>     ;call   delay1_4ms
                            <1>
00014529 C3                 <1>     retn
                            <1>
                            <1> ac97_reset:
                            <1>     ; 27/05/2024 - TRDOS 386 v2.0.8
                            <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
                            <1>     ; 10/06/2017
                            <1>     ; 29/05/2017
                            <1>     ; 28/05/2017
0001452A E8AEFFFFFF         <1>     call    reset_ac97_controller
                            <1>     ; 29/05/2017
                            <1>     ;jmp    reset_ac97_codec
                            <1>     ; 27/05/2024
0001452F E80DF6FFFF         <1>     call    delay_100ms
                            <1> reset_ac97_codec:
                            <1>     ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
00014534 66BA2C00           <1>     mov     dx, GLOB_CNT_REG ; 2Ch
00014538 660315[B6880100]   <1>     add     dx, [NABMBAR]
0001453F ED                 <1>     in      eax, dx
                            <1>
                            <1>     ;test   eax, 2
                            <1>     ; 06/08/2022
00014540 A802               <1>     test    al, 2
00014542 7407               <1>     jz      short _r_ac97codec_cold
                            <1>
00014544 E80F000000         <1>     call    warm_ac97codec_reset
00014549 7308               <1>     jnc     short _r_ac97codec_ok
                            <1> _r_ac97codec_cold:
0001454B E83B000000         <1>         call    cold_ac97codec_reset
00014550 7301               <1>         jnc     short _r_ac97codec_ok
                            <1>
                            <1>     ; 16/04/2017
                            <1>         ;xor       eax, eax       ; timeout error
                            <1>             ;stc
00014552 C3                 <1>     retn
                            <1>
                            <1> _r_ac97codec_ok:
00014553 31C0               <1>         xor     eax, eax
                            <1>         ;mov       al, VIA_ACLINK_C00_READY ; 1
```

```
3865 00014555 FEC0                <1>         inc al
3866 00014557 C3                  <1>     retn
3867                              <1>
3868                              <1> warm_ac97codec_reset:
3869                              <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
3870                              <1>     ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
3871                              <1>         ;mov       eax, 6
3872                              <1>     ; 06/08/2022
3873 00014558 29C0                <1>     sub     eax, eax
3874 0001455A B006                <1>     mov     al, 6
3875 0001455C 66BA2C00            <1>     mov     dx, GLOB_CNT_REG ; 2Ch
3876 00014560 660315[B6880100]    <1>     add     dx, [NABMBAR]
3877 00014567 EF                  <1>     out     dx, eax
3878                              <1>
3879                              <1>     ;mov    ecx, 10 ; total 1s
3880                              <1>     ; 06/08/2022
3881 00014568 31C9                <1>     xor     ecx, ecx
3882 0001456A B10A                <1>     mov     cl, 10
3883                              <1> _warm_ac97c_rst_wait:
3884 0001456C 51                  <1>     push    ecx
3885 0001456D E8CFF5FFFF          <1>     call    delay_100ms
3886 00014572 59                  <1>     pop     ecx
3887                              <1>
3888 00014573 66BA3000            <1>     mov     dx, GLOB_STS_REG ; 30h
3889 00014577 660315[B6880100]    <1>     add     dx, [NABMBAR]
3890 0001457E ED                  <1>     in      eax, dx
3891                              <1>
3892 0001457F A900030010          <1>     test    eax, CTRL_ST_CREADY
3893 00014584 7504                <1>     jnz     short _warm_ac97c_rst_ok
3894                              <1>
3895 00014586 49                  <1>         dec     ecx
3896 00014587 75E3                <1>         jnz     short _warm_ac97c_rst_wait
3897                              <1>
3898                              <1> _warm_ac97c_rst_fail:
3899 00014589 F9                  <1>         stc
3900                              <1> _warm_ac97c_rst_ok:
3901 0001458A C3                  <1>     retn
3902                              <1>
3903                              <1> cold_ac97codec_reset:
3904                              <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
3905                              <1>     ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
3906                              <1>         ;mov       eax, 2
3907                              <1>     ; 06/08/2022
3908 0001458B 31C0                <1>     xor     eax, eax
3909 0001458D B002                <1>     mov     al, 2
3910 0001458F 66BA2C00            <1>     mov     dx, GLOB_CNT_REG ; 2Ch
3911 00014593 660315[B6880100]    <1>     add     dx, [NABMBAR]
3912 0001459A EF                  <1>     out     dx, eax
3913                              <1>
3914 0001459B E8A1F5FFFF          <1>     call    delay_100ms    ; wait 100 ms
3915 000145A0 E89CF5FFFF          <1>     call    delay_100ms    ; wait 100 ms
3916 000145A5 E897F5FFFF          <1>     call    delay_100ms    ; wait 100 ms
3917 000145AA E892F5FFFF          <1>     call    delay_100ms    ; wait 100 ms
3918                              <1>
3919                              <1>     ;mov    ecx, 16 ; total 20*100 ms = 2s
3920                              <1>     ; 06/08/2022
3921 000145AF 31C9                <1>     xor     ecx, ecx
3922 000145B1 B110                <1>     mov     cl, 16
3923                              <1> _cold_ac97c_rst_wait:
3924 000145B3 66BA3000            <1>     mov     dx, GLOB_STS_REG ; 30h
3925 000145B7 660315[B6880100]    <1>     add     dx, [NABMBAR]
3926 000145BE ED                  <1>     in      eax, dx
3927                              <1>
3928 000145BF A900030010          <1>     test    eax, CTRL_ST_CREADY
3929 000145C4 750B                <1>     jnz     short _cold_ac97c_rst_ok
3930                              <1>
3931 000145C6 51                  <1>     push    ecx
3932 000145C7 E875F5FFFF          <1>     call    delay_100ms
3933 000145CC 59                  <1>     pop     ecx
3934                              <1>
3935 000145CD 49                  <1>         dec     ecx
3936 000145CE 75E3                <1>         jnz     short _cold_ac97c_rst_wait
3937                              <1>
3938                              <1> _cold_ac97c_rst_fail:
3939 000145D0 F9                  <1>         stc
3940                              <1> _cold_ac97c_rst_ok:
3941 000145D1 C3                  <1>     retn
3942                              <1>
3943                              <1> sb16_current_sound_data:
3944                              <1>     ; 05/06/2024
3945                              <1>     ; 04/06/2024 - TRDOS 386 v2.0.8
3946                              <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
3947                              <1>     ; 20/08/2017
3948                              <1>     ; 24/06/2017
3949                              <1>     ; 22/06/2017
3950                              <1>     ; get current sound (PCM out) data for graphics
3951                              <1>     ; (for Sound Blaster 16)
3952                              <1>     ; ebx = Physical address (on page boundary)
3953                              <1>     ; ecx = Byte count
3954                              <1>     ; [audio_buff_size]
3955                              <1>
3956                              <1>     ;;mov  edi, [audio_buff_size]
3957                              <1>     ;mov    edi, [audio_dmabuff_size]
3958                              <1>     ;mov    esi, [audio_dma_buff]
3959                              <1>
3960 000145D2 39CF                <1>     cmp     edi, ecx
3961 000145D4 7302                <1>     jnb     short sb16_gcd_0
3962 000145D6 89F9                <1>     mov     ecx, edi
3963                              <1> sb16_gcd_0:
3964                              <1>     ; 06/08/2022
3965 000145D8 31C0                <1>     xor     eax, eax
3966                              <1>     ; 20/08/2017
3967 000145DA 803D[E4880100]10    <1>     cmp     byte [audio_bps], 16
3968 000145E1 750C                <1>     jne     short sb16_gcd_1 ; 8 bit DMA channel
3969 000145E3 E4C6                <1>     in      al, 0C6h ; DMA channel 5 count register
3970                              <1>     ;mov    dl, al
3971                              <1>     ; 06/08/2022
3972 000145E5 88C4                <1>     mov     ah, al
3973 000145E7 E4C6                <1>     in      al, 0C6h
3974                              <1>     ;mov    dh, al
3975                              <1>     ;movzx  eax, dx
3976                              <1>     ; 06/08/2022
3977 000145E9 86C4                <1>     xchg    ah, al
3978 000145EB D1E0                <1>     shl     eax, 1 ; word count -> byte count
3979 000145ED EB4C                <1>     jmp     short sb16_gcd_2
3980                              <1> sb16_gcd_1:
3981 000145EF E403                <1>     in      al, 03h ; DMA channel 1 count register
3982                              <1>     ;mov    dl, al
3983                              <1>     ; 06/08/2022
3984 000145F1 88C4                <1>     mov     ah, al
3985 000145F3 E403                <1>     in      al, 03h
3986                              <1>     ;mov    dh, al
3987                              <1>     ;movzx  eax, dx
3988                              <1>     ; 06/08/2022
```

```
                                <1>     xchg    ah, al
000145F5 86C4                   <1>     jmp     short sb16_gcd_2
000145F7 EB42                   <1> ;sb16_gcd_2:
                                <1> ;   cmp     eax, ecx
                                <1> ;   jnb     short sb16_gcd_3
                                <1> ;   ; remain count < graphics bytes
                                <1> ;   mov     eax, ecx ; fix remain count to data size
                                <1> ;sb16_gcd_3:
                                <1> ;   sub     edi, eax
                                <1> ;   jna     short sb16_gcd_4
                                <1> ;   add     esi, edi ; dma buffer offset
                                <1> ;sb16_gcd_4:
                                <1> ;   mov     edi, ebx ; buffer address (for graphics)
                                <1> ;   mov     [u.r0], ecx
                                <1> ;   rep     movsb
                                <1> ;   retn
                                <1>
                                <1> get_current_sound_data:
                                <1>     ; 05/06/2024
                                <1>     ; 04/06/2024
                                <1>     ; 24/06/2017
                                <1>     ; 22/06/2017
                                <1>     ; get current sound (PCM out) data for graphics
                                <1>     ;
                                <1>     ; ebx = Physical address (on page boundary)
                                <1>     ; ecx = Byte count
                                <1>     ; [audio_buff_size]
                                <1>
                                <1>     ;mov   edi, [audio_buff_size]
                                <1>     ;;;
                                <1>     ; 04/06/2024
                                <1>     ;mov   edi, [audio_dmabuff_size]
                                <1>     ; 05/06/2024
                                <1>     ;mov   edi, [audio_buff_size]
000145F9 8B3D[D8880100]         <1>     mov     edi, [dma_hbuff_size]
                                <1>     ;;;
000145FF 8B35[D0880100]         <1>     mov     esi, [audio_dma_buff]
                                <1>
                                <1>     ;;;
                                <1>     ; 04/06/2024
                                <1>     ;cmp   byte [audio_device], 2
                                <1>     ;jb    short sb16_current_sound_data ; = 1
                                <1>     ;shr   edi, 1
                                <1>
                                <1>     ; 04/06/2024
00014605 803D[B1880100]01       <1>     cmp     byte [audio_device], 1
0001460C 7704                   <1>     ja      short gcd_1
0001460E D1E7                   <1>     shl     edi, 1 ; [audio_dmabuff_size] = 2 * edi
                                <1>     ; 05/06/2024
                                <1>     ; edi = [dma_hbuff_size] * 2
00014610 EBC0                   <1>     jmp     short sb16_current_sound_data
                                <1> gcd_1:
                                <1>     ;;;
                                <1>
                                <1> ; 04/06/2024
                                <1> %if 0
                                <1>     cmp     edi, ecx
                                <1>     jnb     short gcd_0
                                <1>     mov     ecx, edi
                                <1> gcd_0:
                                <1> %endif
00014612 803D[B1880100]03       <1>     cmp     byte [audio_device], 3
00014619 7237                   <1>     jb      short ac97_current_sound_data ; = 2
                                <1>     ; = 3
                                <1> vt8233_current_sound_data:
                                <1>     ; 05/06/2024
                                <1>     ; 04/06/2024 - TRDOS 386 v2.0.8
                                <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
                                <1>     ; 22/06/2017
                                <1>     ; 21/06/2017
                                <1>     ; get current sound (PCM out) data for graphics
                                <1>     ; (for VT 8233, VT 8237R)
                                <1>     ; ebx = Physical address (on page boundary)
                                <1>     ; ecx = Byte count
                                <1>     ; [audio_buff_size]
                                <1>
                                <1>     ;;mov  edi, [audio_buff_size]
                                <1>     ;mov   edi, [audio_dmabuff_size]
                                <1>     ;mov   esi, [audio_dma_buff]
                                <1>     ;shr   edi, 1
                                <1>     ;cmp   edi, ecx
                                <1>     ;jnb   short vt8233_gcd_1
                                <1>     ;mov   ecx, edi
                                <1>
                                <1>     ;;;
                                <1>     ; 04/06/2024
                                <1>     ; edi = [audio_buff_size]
                                <1>     ; 05/06/2024
                                <1>     ;and   di, ~1  ; word alignment
                                <1>     ; edi = [dma_hbuff_size]
0001461B 39CF                   <1>     cmp     edi, ecx
0001461D 7302                   <1>     jnb     short vt8233_gcd_1
0001461F 89F9                   <1>     mov     ecx, edi
                                <1>     ;;;
                                <1>
                                <1> vt8233_gcd_1:
                                <1>     ;mov   edx, VIA_REG_OFFSET_CURR_COUNT
                                <1>     ; 06/08/2022
00014621 31D2                   <1>     xor     edx, edx
00014623 B20C                   <1>     mov     dl, VIA_REG_OFFSET_CURR_COUNT
00014625 E877F5FFFF             <1>     call    ctrl_io_r32
0001462A 89C2                   <1>     mov     edx, eax ; remain count (bits 23-0),
                                <1>                      ; SGD index (bits 31-24)
0001462C 81E200000001           <1>     and     edx, 1000000h ; SGD index (0 = 1st half)
00014632 7402                   <1>     jz      short vt8233_gcd_2
                                <1>     ; the second half of DMA buffer
00014634 01FE                   <1>     add     esi, edi
                                <1> vt8233_gcd_2:
00014636 25FFFFFF00             <1>     and     eax, 0FFFFFFh ; bits 23-0
                                <1> ac97_gcd_2:
                                <1> sb16_gcd_2:
0001463B 39C8                   <1>     cmp     eax, ecx
0001463D 7302                   <1>     jnb     short vt8233_gcd_3
                                <1>     ; remain count < graphics bytes
0001463F 89C8                   <1>     mov     eax, ecx ; fix remain count to data size
                                <1> vt8233_gcd_3:
00014641 29C7                   <1>     sub     edi, eax
00014643 7602                   <1>     jna     short vt8233_gcd_4
00014645 01FE                   <1>     add     esi, edi ; dma buffer offset
                                <1> vt8233_gcd_4:
00014647 89DF                   <1>     mov     edi, ebx ; buffer address (for graphics)
00014649 890D[348E0100]         <1>     mov     [u.r0], ecx
0001464F F3A4                   <1>     rep     movsb
                                <1> vt8233_gcd_5:
```

```
4113 00014651 C3                  <1>     retn
4114                              <1>
4115                              <1> ac97_current_sound_data:
4116                              <1>     ; 05/06/2024
4117                              <1>     ; 04/06/2024 - TRDOS 386 v2.0.8
4118                              <1>     ; 23/06/2017
4119                              <1>     ; 22/06/2017
4120                              <1>     ; get current sound (PCM out) data for graphics
4121                              <1>     ; (for AC'97, ICH)
4122                              <1>     ; ebx = Physical address (on page boundary)
4123                              <1>     ; ecx = Byte count
4124                              <1>     ; [audio_buff_size]
4125                              <1>
4126                              <1>     ;;mov   edi, [audio_buff_size]
4127                              <1>     ;mov    edi, [audio_dmabuff_size]
4128                              <1>     ;mov    esi, [audio_dma_buff]
4129                              <1>     ;shr    edi, 1
4130                              <1>     ;cmp    edi, ecx
4131                              <1>     ;jnb    short ac97_gcd_0
4132                              <1>     ;mov    ecx, edi
4133                              <1>
4134                              <1>     ;;;
4135                              <1>     ; 04/06/2024
4136                              <1>     ; edi = [audio_buff_size]
4137                              <1>     ; 05/06/2024
4138                              <1>     ;and    di, ~7  ; 8 byte alignment
4139                              <1>     ; edi = [dma_hbuff_size]
4140 00014652 39CF                <1>     cmp     edi, ecx
4141 00014654 7302                <1>     jnb     short ac97_gcd_0
4142 00014656 89F9                <1>     mov     ecx, edi
4143                              <1>     ;;;
4144                              <1>
4145                              <1> ac97_gcd_0:
4146 00014658 66BA1400            <1>     mov     dx, PO_CIV_REG ; Position In Current Buff Reg
4147 0001465C 660315[B6880100]    <1>     add     dx, [NABMBAR]
4148 00014663 EC                  <1>     in      al, dx ; current index value
4149 00014664 A801                <1>     test    al, 1
4150 00014666 7402                <1>     jz      short ac97_gcd_1
4151 00014668 01FE                <1>     add     esi, edi
4152                              <1> ac97_gcd_1:
4153 0001466A 31C0                <1>     xor     eax, eax
4154 0001466C 66BA1800            <1>     mov     dx, PO_PICB_REG ; Position In Current Buff Reg
4155 00014670 660315[B6880100]    <1>     add     dx, [NABMBAR]
4156                              <1>     ;in     ax, dx ; remain dwords
4157                              <1>     ;shl    eax, 2 ; remain bytes ; 23/06/2017
4158                              <1>     ;;;
4159                              <1>     ; 04/06/2024
4160 00014677 66ED                <1>     in      ax, dx ; remain words
4161 00014679 D1E0                <1>     shl     eax, 1 ; remain bytes
4162                              <1>     ;;;
4163 0001467B EBBE                <1>     jmp     short ac97_gcd_2
4164                              <1> ;   cmp     eax, ecx
4165                              <1> ;   jnb     short ac97_gcd_2
4166                              <1> ;   ; remain count < graphics bytes
4167                              <1> ;   mov     eax, ecx ; fix remain count to data size
4168                              <1> ;ac97_gcd_2:
4169                              <1> ;   sub     edi, eax
4170                              <1> ;   jna     short ac97_gcd_3
4171                              <1> ;   add     esi, edi ; dma buffer offset
4172                              <1> ;ac97_gcd_3:
4173                              <1> ;   mov     edi, ebx ; buffer address (for graphics)
4174                              <1> ;   mov     [u.r0], ecx
4175                              <1> ;   rep     movsb
4176                              <1> ;   retn
4177                              <1>
4178                              <1> sb16_get_dma_buff_off:
4179                              <1>     ; 05/06/2024
4180                              <1>     ; 04/06/2024 - TRDOS 386 v2.0.8
4181                              <1>     ; 28/10/2017
4182                              <1>     ; 24/06/2017
4183                              <1>     ; 22/06/2017
4184                              <1>     ; get current (PCM OUT DMA buffer) pointer
4185                              <1>     ; (for Sound Blaster 16)
4186                              <1>
4187                              <1>     ;mov    ecx, [audio_dmabuff_size]
4188                              <1>     ;xor    ebx, ebx
4189                              <1>     ;shr    ecx, 1
4190                              <1>
4191                              <1>     ;;;
4192                              <1>     ; 04/06/2024
4193                              <1>     ; ecx = audio buffer size
4194                              <1>     ; 05/06/2024
4195                              <1>     ; ecx = DMA half buffer size
4196 0001467D D1E1                <1>     shl     ecx, 1 ; * 2
4197                              <1>     ; ecx = DMA buffer size
4198                              <1>     ;;;
4199                              <1>
4200                              <1> sb16_gdmabo_0:
4201                              <1>     ; 28/10/2017
4202 0001467F 803D[E4880100]10    <1>     cmp     byte [audio_bps], 16
4203 00014686 750F                <1>     jne     short sb16_gdmabo_1 ; 8 bit DMA channel
4204                              <1>     ; 16 bit DMA channel
4205 00014688 E4C6                <1>     in      al, 0C6h ; DMA channel 5 count register
4206 0001468A 88C2                <1>     mov     dl, al
4207 0001468C E4C6                <1>     in      al, 0C6h
4208 0001468E 88C6                <1>     mov     dh, al
4209 00014690 0FB7C2              <1>     movzx   eax, dx
4210 00014693 D1E0                <1>     shl     eax, 1 ; word count -> byte count
4211 00014695 EB3A                <1>     jmp     short sb16_gdmabo_2
4212                              <1> sb16_gdmabo_1:
4213 00014697 E403                <1>     in      al, 03h ; DMA channel 1 count register
4214 00014699 88C2                <1>     mov     dl, al
4215 0001469B E403                <1>     in      al, 03h
4216 0001469D 88C6                <1>     mov     dh, al
4217 0001469F 0FB7C2              <1>     movzx   eax, dx
4218 000146A2 EB2D                <1>     jmp     short sb16_gdmabo_2
4219                              <1>
4220                              <1> get_dma_buffer_offset:
4221                              <1>     ; 05/06/2024
4222                              <1>     ; 04/06/2024
4223                              <1>     ; 24/06/2017
4224                              <1>     ; 22/06/2017
4225                              <1>     ; get current sound (PCM out) data for graphics
4226                              <1>     ;
4227                              <1>     ; ebx = Physical address (on page boundary)
4228                              <1>     ; ecx = Byte count
4229                              <1>     ; [audio_buff_size]
4230                              <1>
4231                              <1>     ;mov    ecx, [audio_dmabuff_size]
4232                              <1>     ;;;
4233                              <1>     ; 04/06/2024
4234                              <1>     ;mov    ecx, [audio_buff_size]
4235                              <1>     ; 05/06/2024
4236 000146A4 8B0D[D8880100]      <1>     mov     ecx, [dma_hbuff_size]
```

```
                                <1>     ;;;
000146AA 31DB                   <1>     xor     ebx, ebx
                                <1> gdmabo_0:
000146AC 803D[B1880100]02       <1>     cmp     byte [audio_device], 2
000146B3 72C8                   <1>     jb      short sb16_get_dma_buff_off
000146B5 7427                   <1>     je      short ac97_get_dma_buff_off
                                <1>
                                <1> vt8233_get_dma_buff_off:
                                <1>     ; 05/06/2024
                                <1>     ; 04/06/2024 - TRDOS 386 v2.0.8
                                <1>     ; 06/08/2022 - TRDOS 386 v2.0.5
                                <1>     ; 24/06/2017
                                <1>     ; 22/06/2017
                                <1>     ; get current (PCM OUT DMA buffer) pointer
                                <1>     ; (for VT 8233, VT 8237R)
                                <1>
                                <1>     ;mov    ecx, [audio_dmabuff_size]
                                <1>     ;xor    ebx, ebx
                                <1>     ;;;
                                <1>     ; 04/06/2024
                                <1>     ;shr    ecx, 1
                                <1>     ; 05/06/2024
                                <1>     ;and    cl, ~1
                                <1>     ; ecx = DMA Half Buffer Size (word aligned)
                                <1>     ;;;
                                <1> vt8233_gdmabo_0:
                                <1>     ;mov    edx, VIA_REG_OFFSET_CURR_COUNT
                                <1>     ; 06/08/2022
000146B7 31D2                   <1>     xor     edx, edx
000146B9 B20C                   <1>     mov     dl, VIA_REG_OFFSET_CURR_COUNT
000146BB E8E1F4FFFF             <1>     call    ctrl_io_r32
000146C0 89C2                   <1>     mov     edx, eax ; remain count (bits 23-0),
                                <1>                      ; SGD index (bits 31-24)
000146C2 81E200000001           <1>     and     edx, 1000000h ; SGD index (0 = 1st half)
000146C8 7402                   <1>     jz      short vt8233_gdmabo_1
                                <1>     ; the second half of DMA buffer
000146CA 89CB                   <1>     mov     ebx, ecx
                                <1> vt8233_gdmabo_1:
000146CC 25FFFFFF00             <1>     and     eax, 0FFFFFFh ; bits 23-0
                                <1> sb16_gdmabo_2:
                                <1> ac97_gdmabo_2:
000146D1 29C1                   <1>     sub     ecx, eax
000146D3 7602                   <1>     jna     short vt8233_gdmabo_2
000146D5 01CB                   <1>     add     ebx, ecx ; dma buffer offset
                                <1> vt8233_gdmabo_2:
000146D7 891D[348E0100]         <1>     mov     [u.r0], ebx
000146DD C3                     <1>     retn
                                <1>
                                <1> ac97_get_dma_buff_off:
                                <1>     ; 06/06/2024
                                <1>     ; 05/06/2024
                                <1>     ; 04/06/2024 - TRDOS 386 v2.0.8
                                <1>     ; 24/06/2017
                                <1>     ; 22/06/2017
                                <1>     ; get current (PCM OUT DMA buffer) pointer
                                <1>     ; (for AC'97, ICH)
                                <1>     ; ebx = Physical address (on page boundary)
                                <1>     ; ecx = Byte count
                                <1>     ; [audio_buff_size]
                                <1>
                                <1>     ;mov    ecx, [audio_dmabuff_size]
                                <1>     ;xor    ebx, ebx
                                <1>     ;;;
                                <1>     ; 04/06/2024
                                <1>     ;shr    ecx, 1
                                <1>     ; 05/06/2024
                                <1>     ;and    cl, ~7 ; (truncate bytes if out of 8x)
                                <1>     ; ecx = DMA Half Buffer Size (8 byte aligned)
                                <1>     ;;;
                                <1> ac97_gdmabo_0:
000146DE 66BA1400               <1>     mov     dx, PO_CIV_REG ; Position In Current Buff Reg
000146E2 660315[B6880100]       <1>     add     dx, [NABMBAR]
000146E9 EC                     <1>     in      al, dx ; current index value
000146EA A801                   <1>     test    al, 1
000146EC 7402                   <1>     jz      short ac97_gdmabo_1
000146EE 89CB                   <1>     mov     ebx, ecx
                                <1> ac97_gdmabo_1:
000146F0 31C0                   <1>     xor     eax, eax
000146F2 66BA1800               <1>     mov     dx, PO_PICB_REG ; Position In Current Buff Reg
000146F6 660315[B6880100]       <1>     add     dx, [NABMBAR]
000146FD 66ED                   <1>     in      ax, dx ; remain words (samples)
                                <1>     ;;;
                                <1>     ; 06/06/2024
                                <1>     ; audio samples are counted as words
                                <1>     ; 04/06/2024 (BugFix)
000146FF D1E0                   <1>     shl     eax, 1 ; remain bytes
                                <1>     ;;;
00014701 EBCE                   <1>     jmp     short ac97_gdmabo_2
                                <1>
                                <1> ac97_codec_info:
                                <1>     ; 06/06/2024
                                <1>     ; 05/06/2024 - TRDOS 386 v2.0.8
                                <1>     ; 19/11/2023
                                <1>     ; ENTRY: none
                                <1>     ; RETURN:
                                <1>     ;       eax = Extended Audio ID (MX28) in ax
                                <1>     ;        ax bit 0 - VRA bit
                                <1>     ;        hw of eax = PCM output sample rate (48000)
                                <1>     ;       ebx = VENDOR ID 1, VENDOR ID 2 (bx)
                                <1>     ;       cf = 1 -> error
                                <1>
                                <1>     ; 06/06/2024
                                <1>     ;;;
00014703 66BA3000               <1>     mov     dx, GLOB_STS_REG ; 30h
00014707 660315[B6880100]       <1>     add     dx, [NABMBAR]
0001470E ED                     <1>     in      eax, dx
                                <1>
                                <1>     ;cmp    eax, 0FFFFFFFFh ; -1
0001470F 40                     <1>     inc     eax
00014710 7408                   <1>     jz      short ac97_c_inf_err
00014712 48                     <1>     dec     eax
                                <1>
00014713 2500030010             <1>     and     eax, CTRL_ST_CREADY
00014718 7502                   <1>     jnz     short ac97_c_inf_1
                                <1>
                                <1> ac97_c_inf_err:
0001471A F9                     <1>     stc
0001471B C3                     <1>     retn
                                <1>     ;;;
                                <1>
                                <1> ac97_c_inf_1:
0001471C 668B15[B4880100]       <1>     mov     dx, [NAMBAR]
00014723 6683C22C               <1>     add     dx, CODEC_PCM_FRONT_DACRATE_REG
00014727 66ED                   <1>     in      ax, dx  ; PCM Output Sample Rate
```

```
4361                                    <1>                     ; 48000 Hz
4362 00014729 C1E010                    <1>     shl     eax, 16
4363 0001472C E810F4FFFF                <1>     call    delay_100ms
4364 00014731 668B15[B4880100]          <1>     mov     dx, [NAMBAR]
4365 00014738 6683C228                  <1>     add     dx, CODEC_EXT_AUDIO_REG ; 28h
4366 0001473C 66ED                      <1>     in      ax, dx
4367 0001473E 50                        <1>     push    eax
4368 0001473F E8FDF3FFFF                <1>     call    delay_100ms
4369 00014744 668B15[B4880100]          <1>     mov     dx, [NAMBAR]
4370 0001474B 6683C27C                  <1>     add     dx, CODEC_VENDOR_ID1
4371 0001474F 66ED                      <1>     in      ax, dx
4372 00014751 C1E010                    <1>     shl     eax, 16
4373 00014754 668B15[B4880100]          <1>     mov     dx, [NAMBAR]
4374 0001475B 6683C27E                  <1>     add     dx, CODEC_VENDOR_ID2
4375 0001475F 66ED                      <1>     in      ax, dx
4376 00014761 89C3                      <1>     mov     ebx, eax
4377 00014763 58                        <1>     pop     eax
4378                                    <1>     ;;;;
4379                                    <1>     ; 06/06/2024 (temporary!?)
4380                                    <1>     ; (ALC850 & ALC655 BugFix)
4381 00014764 83FBFF                    <1>     cmp     ebx, 0FFFFFFFFh ; -1
4382 00014767 74B1                      <1>     je      short ac97_c_inf_err ; invalid
4383 00014769 09DB                      <1>     or      ebx, ebx
4384 0001476B 74AD                      <1>     jz      short ac97_c_inf_err ; invalid
4385 0001476D 81FB90474C41              <1>     cmp     ebx, 414C4790h ; ALC850
4386                                    <1>     ;je     short ac97_c_inf_2
4387                                    <1>     ;cmp    ebx, 414C4760h ; ALC655
4388 00014773 7505                      <1>     jne     short ac97_c_inf_3
4389                                    <1>     ;mov    eax, 0BB8009C6h ; AC655 default (Read only)
4390                                    <1>     ;retn
4391                                    <1> ac97_c_inf_2:
4392                                    <1>     ; ref: ALC850 & ALC655 data sheet
4393 00014775 B8C40980BB                <1>     mov     eax, 0BB8009C4h ; AC850 default (Read only)
4394                                    <1> ac97_c_inf_3:
4395                                    <1>     ;;;;
4396 0001477A C3                        <1>     retn
4397                                    <1>
4398                                    <1> ; 19/11/2023
4399                                    <1> ; Valid ICH device IDs
4400                                    <1>
4401                                    <1> valid_ids:
4402 0001477B 86801524                  <1> dd  (ICH_DID << 16) + INTEL_VID    ; 8086h:2415h
4403 0001477F 86802524                  <1> dd  (ICH0_DID << 16) + INTEL_VID   ; 8086h:2425h
4404 00014783 86804524                  <1> dd  (ICH2_DID << 16) + INTEL_VID   ; 8086h:2445h
4405 00014787 86808524                  <1> dd  (ICH3_DID << 16) + INTEL_VID   ; 8086h:2485h
4406 0001478B 8680C524                  <1> dd  (ICH4_DID << 16) + INTEL_VID   ; 8086h:24C5h
4407 0001478F 8680D524                  <1> dd  (ICH5_DID << 16) + INTEL_VID   ; 8086h:24D5h
4408 00014793 86806E26                  <1> dd  (ICH6_DID << 16) + INTEL_VID   ; 8086h:266Eh
4409 00014797 8680A625                  <1> dd  (ESB6300_DID << 16) + INTEL_VID  ; 8086h:25A6h
4410 0001479B 86809826                  <1> dd  (ESB631X_DID << 16) + INTEL_VID  ; 8086h:2698h
4411 0001479F 8680DE27                  <1> dd  (ICH7_DID << 16) + INTEL_VID   ; 8086h:27DEh
4412 000147A3 86809571                  <1> dd  (MX82440_DID << 16) + INTEL_VID  ; 8086h:7195h
4413 000147A7 39101270                  <1> dd  (SI7012_DID << 16)  + SIS_VID  ; 1039h:7012h
4414 000147AB DE10B101                  <1> dd  (NFORCE_DID << 16)  + NVIDIA_VID ; 10DEh:01B1h
4415 000147AF DE106A00                  <1> dd  (NFORCE2_DID << 16) + NVIDIA_VID ; 10DEh:006Ah
4416 000147B3 22106D74                  <1> dd  (AMD8111_DID << 16) + AMD_VID          ; 1022h:746Dh
4417 000147B7 22104574                  <1> dd  (AMD768_DID << 16)  + AMD_VID          ; 1022h:7445h
4418 000147BB DE105900                  <1> dd  (CK804_DID << 16) + NVIDIA_VID         ; 10DEh:0059h
4419 000147BF DE103A00                  <1> dd  (MCP04_DID << 16) + NVIDIA_VID         ; 10DEh:003Ah
4420 000147C3 DE108A00                  <1> dd  (CK8_DID << 16) + NVIDIA_VID   ; 1022h:008Ah
4421 000147C7 DE10DA00                  <1> dd  (NFORCE3_DID << 16) + NVIDIA_VID ; 10DEh:00DAh
4422 000147CB DE10EA00                  <1> dd  (CK8S_DID << 16) + NVIDIA_VID  ; 10DEh:00EAh
4423                                    <1>
4424                                    <1> valid_id_count:    equ ($ - valid_ids)>>2 ; 19/11/2023
3786
3787 000147CF 90                        align 4
3788
3789                                    %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 000147D0 00000000000000000-        <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
  31 000147D9 00000000000000    <1>
  32 000147E0 0000000000000002A-        <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
  32 000147E9 2A2A2A2A2A2A      <1>
  33 000147F0 2A2A2A2A2A2A2A2A2A-       <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
  33 000147F9 2A2A2A2A2A2A      <1>
  34 00014800 2A2A2A2A2A2A2A2A2A-       <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
  34 00014809 2A2A2A2A2A2A      <1>
  35 00014810 2A2A2A2A2A2A2A2A3F-       <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh
  35 00014819 3F3F3F3F3F3F      <1>
  36 00014820 3F3F3F3F3F3F3F3F3F-       <1>     db  03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh
  36 00014829 3F3F3F3F3F3F      <1>
  37 00014830 00000000000000000-        <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
  37 00014839 00000000000000    <1>
  38 00014840 0000000000000002A-        <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
  38 00014849 2A2A2A2A2A2A      <1>
  39 00014850 2A2A2A2A2A2A2A2A2A-       <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
  39 00014859 2A2A2A2A2A2A      <1>
  40 00014860 2A2A2A2A2A2A2A2A2A-       <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
  40 00014869 2A2A2A2A2A2A      <1>
  41 00014870 2A2A2A2A2A2A2A2A3F-       <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh
  41 00014879 3F3F3F3F3F3F      <1>
  42 00014880 3F3F3F3F3F3F3F3F3F-       <1>     db  03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh
  42 00014889 3F3F3F3F3F3F      <1>
  43                                    <1> palette1: ; (63+1)*3
  44 00014890 00000000002A002A00-       <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah
```

```
00014899 002A2A2A00002A      <1>
000148A0 002A2A15002A2A2A00- <1>     db  000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah
000148A9 000000002A002A      <1>
000148B0 00002A2A2A00002A00- <1>     db  000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah, 000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah
000148B9 2A2A15002A2A2A      <1>
000148C0 15151515153F153F15- <1>     db  015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh, 015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh
000148C9 153F3F3F15153F      <1>
000148D0 153F3F3F153F3F3F15- <1>     db  015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh
000148D9 151515153F153F      <1>
000148E0 15153F3F3F15153F15- <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh
000148E9 3F3F3F153F3F3F      <1>
000148F0 00000000002A002A00- <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah
000148F9 002A2A2A00002A      <1>
00014900 002A2A15002A2A2A00- <1>     db  000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah
00014909 000000002A002A      <1>
00014910 00002A2A2A00002A00- <1>     db  000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah, 000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah
00014919 2A2A15002A2A2A      <1>
00014920 15151515153F153F15- <1>     db  015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh, 015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh
00014929 153F3F3F15153F      <1>
00014930 153F3F3F153F3F3F15- <1>     db  015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh
00014939 151515153F153F      <1>
00014940 15153F3F3F15153F15- <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh
00014949 3F3F3F153F3F3F      <1>
                             <1> palette2: ; (63+1)*3
00014950 00000000002A002A00- <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah
00014959 002A2A2A00002A      <1>
00014960 002A2A2A002A2A2A00- <1>     db  000h, 02ah, 02ah, 02ah, 000h, 02ah, 02ah, 02ah, 000h, 000h, 015h, 000h, 000h, 03fh, 000h, 02ah
00014969 001500003F002A      <1>
00014970 15002A3F2A00152A00- <1>     db  015h, 000h, 02ah, 03fh, 02ah, 000h, 015h, 02ah, 000h, 03fh, 02ah, 02ah, 015h, 02ah, 02ah, 03fh
00014979 3F2A2A152A2A3F      <1>
00014980 00150000152A003F00- <1>     db  000h, 015h, 000h, 000h, 015h, 02ah, 000h, 03fh, 000h, 000h, 03fh, 02ah, 02ah, 015h, 000h, 02ah
00014989 003F2A2A15002A      <1>
00014990 152A2A3F002A3F2A00- <1>     db  015h, 02ah, 02ah, 03fh, 000h, 02ah, 03fh, 02ah, 000h, 015h, 015h, 000h, 015h, 03fh, 000h, 03fh
00014999 151500153F003F      <1>
000149A0 15003F3F2A15152A15- <1>     db  015h, 000h, 03fh, 03fh, 02ah, 015h, 015h, 02ah, 015h, 03fh, 02ah, 03fh, 015h, 02ah, 03fh, 03fh
000149A9 3F2A3F152A3F3F      <1>
000149B0 15000015002A152A00- <1>     db  015h, 000h, 000h, 015h, 000h, 02ah, 015h, 02ah, 000h, 015h, 02ah, 02ah, 03fh, 000h, 000h, 03fh
000149B9 152A2A3F00003F      <1>
000149C0 002A3F2A003F2A2A15- <1>     db  000h, 02ah, 03fh, 02ah, 000h, 03fh, 02ah, 02ah, 015h, 000h, 015h, 015h, 000h, 03fh, 015h, 02ah
000149C9 001515003F152A      <1>
000149D0 15152A3F3F00153F00- <1>     db  015h, 015h, 02ah, 03fh, 03fh, 000h, 015h, 03fh, 000h, 03fh, 03fh, 02ah, 015h, 03fh, 02ah, 03fh
000149D9 3F3F2A153F2A3F      <1>
000149E0 15150015152A153F00- <1>     db  015h, 015h, 000h, 015h, 015h, 02ah, 015h, 03fh, 000h, 015h, 03fh, 02ah, 03fh, 015h, 000h, 03fh
000149E9 153F2A3F15003F      <1>
000149F0 152A3F3F003F3F2A15- <1>     db  015h, 02ah, 03fh, 03fh, 000h, 03fh, 03fh, 02ah, 015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh
000149F9 151515153F153F      <1>
00014A00 15153F3F3F15153F15- <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh
00014A09 3F3F3F153F3F3F      <1>
                             <1> palette3: ; 256*3
00014A10 00000000002A002A00- <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah
00014A19 002A2A2A00002A      <1>
00014A20 002A2A15002A2A2A15- <1>     db  000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah, 015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh
00014A29 151515153F153F      <1>
00014A30 15153F3F3F15153F15- <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh
00014A39 3F3F3F153F3F3F      <1>
00014A40 00000005050508080 8- <1>     db  000h, 000h, 000h, 005h, 005h, 005h, 008h, 008h, 008h, 00bh, 00bh, 00bh, 00eh, 00eh, 00eh, 011h
00014A49 0B0B0B0E0E0E11      <1>
00014A50 11111414141818181C- <1>     db  011h, 011h, 014h, 014h, 014h, 018h, 018h, 018h, 01ch, 01ch, 01ch, 020h, 020h, 020h, 024h, 024h
00014A59 1C1C2020202424      <1>
00014A60 242828282D2D2D3232- <1>     db  024h, 028h, 028h, 028h, 02dh, 02dh, 02dh, 032h, 032h, 032h, 038h, 038h, 038h, 03fh, 03fh, 03fh
00014A69 323838383F3F3F      <1>
00014A70 00003F10003F1F003F- <1>     db  000h, 000h, 03fh, 010h, 000h, 03fh, 01fh, 000h, 03fh, 02fh, 000h, 03fh, 03fh, 000h, 03fh, 03fh
00014A79 2F003F3F003F3F      <1>
00014A80 002F3F001F3F00103F- <1>     db  000h, 02fh, 03fh, 000h, 01fh, 03fh, 000h, 010h, 03fh, 000h, 000h, 03fh, 010h, 000h, 03fh, 01fh
00014A89 00003F10003F1F      <1>
00014A90 003F2F003F3F002F3F- <1>     db  000h, 03fh, 02fh, 000h, 03fh, 03fh, 000h, 02fh, 03fh, 000h, 01fh, 03fh, 000h, 010h, 03fh, 000h
00014A99 001F3F00103F00      <1>
00014AA0 003F00003F10003F1F- <1>     db  000h, 03fh, 000h, 000h, 03fh, 010h, 000h, 03fh, 01fh, 000h, 03fh, 02fh, 000h, 03fh, 03fh, 000h
00014AA9 003F2F003F3F00      <1>
00014AB0 2F3F001F3F00103F1F- <1>     db  02fh, 03fh, 000h, 01fh, 03fh, 000h, 010h, 03fh, 01fh, 01fh, 03fh, 027h, 01fh, 03fh, 02fh, 01fh
00014AB9 1F3F271F3F2F1F      <1>
00014AC0 3F371F3F3F1F3F3F1F- <1>     db  03fh, 037h, 01fh, 03fh, 03fh, 01fh, 03fh, 03fh, 01fh, 037h, 03fh, 01fh, 02fh, 03fh, 01fh, 027h
00014AC9 373F1F2F3F1F27      <1>
00014AD0 3F1F1F3F271F3F2F1F- <1>     db  03fh, 01fh, 01fh, 03fh, 027h, 01fh, 03fh, 02fh, 01fh, 03fh, 037h, 01fh, 03fh, 03fh, 01fh, 037h
00014AD9 3F371F3F3F1F37      <1>
00014AE0 3F1F2F3F1F273F1F1F- <1>     db  03fh, 01fh, 02fh, 03fh, 01fh, 027h, 03fh, 01fh, 01fh, 03fh, 01fh, 01fh, 03fh, 027h, 01fh, 03fh
00014AE9 3F1F1F3F271F3F      <1>
00014AF0 2F1F3F371F3F3F1F37- <1>     db  02fh, 01fh, 03fh, 037h, 01fh, 03fh, 03fh, 01fh, 037h, 03fh, 01fh, 02fh, 03fh, 01fh, 027h, 03fh
00014AF9 3F1F2F3F1F273F      <1>
00014B00 2D2D3F312D3F362D3F- <1>     db  02dh, 02dh, 03fh, 031h, 02dh, 03fh, 036h, 02dh, 03fh, 03ah, 02dh, 03fh, 03fh, 02dh, 03fh, 03fh
00014B09 3A2D3F3F2D3F3F      <1>
00014B10 2D3A3F2D363F2D313F- <1>     db  02dh, 03ah, 03fh, 02dh, 036h, 03fh, 02dh, 031h, 03fh, 02dh, 02dh, 03fh, 031h, 02dh, 03fh, 036h
00014B19 2D2D3F312D3F36      <1>
00014B20 2D3F3A2D3F3F2D3A3F- <1>     db  02dh, 03fh, 03ah, 02dh, 03fh, 03fh, 02dh, 03ah, 03fh, 02dh, 036h, 03fh, 02dh, 031h, 03fh, 02dh
00014B29 2D363F2D313F2D      <1>
00014B30 2D3F2D2D3F312D3F36- <1>     db  02dh, 03fh, 02dh, 02dh, 03fh, 031h, 02dh, 03fh, 036h, 02dh, 03fh, 03ah, 02dh, 03fh, 03fh, 02dh
00014B39 2D3F3A2D3F3F2D      <1>
00014B40 3A3F2D363F2D313F00- <1>     db  03ah, 03fh, 02dh, 036h, 03fh, 02dh, 031h, 03fh, 000h, 000h, 01ch, 007h, 000h, 01ch, 00eh, 000h
00014B49 001C07001C0E00      <1>
00014B50 1C15001C1C001C1C00- <1>     db  01ch, 015h, 000h, 01ch, 01ch, 000h, 01ch, 01ch, 000h, 015h, 01ch, 000h, 00eh, 01ch, 000h, 007h
00014B59 151C000E1C0007      <1>
00014B60 1C00001C07001C0E00- <1>     db  01ch, 000h, 000h, 01ch, 007h, 000h, 01ch, 00eh, 000h, 01ch, 015h, 000h, 01ch, 01ch, 000h, 015h
00014B69 1C15001C1C0015      <1>
00014B70 1C000E1C00071C0000- <1>     db  01ch, 000h, 00eh, 01ch, 000h, 007h, 01ch, 000h, 000h, 01ch, 000h, 000h, 01ch, 007h, 000h, 01ch
00014B79 1C00001C07001C      <1>
00014B80 0E001C15001C1C0015- <1>     db  00eh, 000h, 01ch, 015h, 000h, 01ch, 01ch, 000h, 015h, 01ch, 000h, 00eh, 01ch, 000h, 007h, 01ch
00014B89 1C000E1C00071C      <1>
00014B90 0E0E1C110E1C150E1C- <1>     db  00eh, 00eh, 01ch, 011h, 00eh, 01ch, 015h, 00eh, 01ch, 018h, 00eh, 01ch, 01ch, 00eh, 01ch, 01ch
00014B99 180E1C1C0E1C1C      <1>
00014BA0 0E181C0E151C0E111C- <1>     db  00eh, 018h, 01ch, 00eh, 015h, 01ch, 00eh, 011h, 01ch, 00eh, 00eh, 01ch, 011h, 00eh, 01ch, 015h
00014BA9 0E0E1C110E1C15      <1>
00014BB0 0E1C180E1C1C0E181C- <1>     db  00eh, 01ch, 018h, 00eh, 01ch, 01ch, 00eh, 018h, 01ch, 00eh, 015h, 01ch, 00eh, 011h, 01ch, 00eh
00014BB9 0E151C0E111C0E      <1>
00014BC0 0E1C0E0E1C110E1C15- <1>     db  00eh, 01ch, 00eh, 00eh, 01ch, 011h, 00eh, 01ch, 015h, 00eh, 01ch, 018h, 00eh, 01ch, 01ch, 00eh
00014BC9 0E1C180E1C1C0E      <1>
00014BD0 181C0E151C0E111C14- <1>     db  018h, 01ch, 00eh, 015h, 01ch, 00eh, 011h, 01ch, 014h, 014h, 01ch, 016h, 014h, 01ch, 018h, 014h
00014BD9 141C16141C1814      <1>
00014BE0 1C1A141C1C141C1C14- <1>     db  01ch, 01ah, 014h, 01ch, 01ch, 014h, 01ch, 01ch, 014h, 01ah, 01ch, 014h, 018h, 01ch, 014h, 016h
00014BE9 1A1C14181C1416      <1>
00014BF0 1C14141C16141C1814- <1>     db  01ch, 014h, 014h, 01ch, 016h, 014h, 01ch, 018h, 014h, 01ch, 01ah, 014h, 01ch, 01ch, 014h, 01ah
00014BF9 1C1A141C1C141A      <1>
00014C00 1C14181C14161C1414- <1>     db  01ch, 014h, 018h, 01ch, 014h, 016h, 01ch, 014h, 014h, 01ch, 014h, 014h, 01ch, 016h, 014h, 01ch
00014C09 1C14141C16141C      <1>
00014C10 18141C1A141C1C141A- <1>     db  018h, 014h, 01ch, 01ah, 014h, 01ch, 01ch, 014h, 01ah, 01ch, 014h, 018h, 01ch, 014h, 016h, 01ch
00014C19 1C14181C14161C      <1>
00014C20 0000100400100800 10- <1>     db  000h, 000h, 010h, 004h, 000h, 010h, 008h, 000h, 010h, 00ch, 000h, 010h, 010h, 000h, 010h, 010h
00014C29 0C001010001010      <1>
00014C30 000C10000810000410- <1>     db  000h, 00ch, 010h, 000h, 008h, 010h, 000h, 004h, 010h, 000h, 000h, 010h, 004h, 000h, 010h, 008h
00014C39 00001004001008      <1>
00014C40 00100C001010000C10- <1>     db  000h, 010h, 00ch, 000h, 010h, 010h, 000h, 00ch, 010h, 000h, 008h, 010h, 000h, 004h, 010h, 000h
00014C49 00081000041000      <1>
00014C50 0010000010040010 08- <1>     db  000h, 010h, 000h, 000h, 010h, 004h, 000h, 010h, 008h, 000h, 010h, 00ch, 000h, 010h, 010h, 000h
00014C59 00100C00101000      <1>
00014C60 0C1000081000041008- <1>     db  00ch, 010h, 000h, 008h, 010h, 000h, 004h, 010h, 008h, 008h, 010h, 00ah, 008h, 010h, 00ch, 008h
```

```
00014C69 08100A08100C08      <1>
00014C70 100E08101008101008- <1>     db  010h, 00eh, 008h, 010h, 010h, 008h, 010h, 010h, 008h, 00eh, 010h, 008h, 00ch, 010h, 008h, 00ah
00014C79 0E10080C10080A      <1>
00014C80 100808100A08100C08- <1>     db  010h, 008h, 008h, 010h, 00ah, 008h, 010h, 00ch, 008h, 010h, 00eh, 008h, 010h, 010h, 008h, 00eh
00014C89 100E081010080E      <1>
00014C90 10080C10080A100808- <1>     db  010h, 008h, 00ch, 010h, 008h, 00ah, 010h, 008h, 008h, 010h, 008h, 008h, 010h, 00ah, 008h, 010h
00014C99 100808100A0810      <1>
00014CA0 0C08100E081010080E- <1>     db  00ch, 008h, 010h, 00eh, 008h, 010h, 010h, 008h, 00eh, 010h, 008h, 00ch, 010h, 008h, 00ah, 010h
00014CA9 10080C10080A10      <1>
00014CB0 0B0B100C0B100D0B10- <1>     db  00bh, 00bh, 010h, 00ch, 00bh, 010h, 00dh, 00bh, 010h, 00fh, 00bh, 010h, 010h, 00bh, 010h, 010h
00014CB9 0F0B10100B1010      <1>
00014CC0 0B0F100B0D100B0C10- <1>     db  00bh, 00fh, 010h, 00bh, 00dh, 010h, 00bh, 00ch, 010h, 00bh, 00bh, 010h, 00ch, 00bh, 010h, 00dh
00014CC9 0B0B100C0B100D      <1>
00014CD0 0B100F0B10100B0F10- <1>     db  00bh, 010h, 00fh, 00bh, 010h, 010h, 00bh, 00fh, 010h, 00bh, 00dh, 010h, 00bh, 00ch, 010h, 00bh
00014CD9 0B0D100B0C100B      <1>
00014CE0 0B100B0B100C0B100D- <1>     db  00bh, 010h, 00bh, 00bh, 010h, 00ch, 00bh, 010h, 00dh, 00bh, 010h, 00fh, 00bh, 010h, 010h, 00bh
00014CE9 0B100F0B10100B      <1>
00014CF0 0F100B0D100B0C1000- <1>     db  00fh, 010h, 00bh, 00dh, 010h, 00bh, 00ch, 010h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
00014CF9 00000000000000      <1>
00014D00 0000000000000000000- <1>    db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
00014D09 00000000000000      <1>
                             <1>
                             <1>
                             <1> ; 04/07/2016
                             <1> ; FONT DATA
                             <1>
                             <1> CRT_CHAR_GEN:
                             <1> vgafont8:
00014D10 00000000000000007E- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 081h, 0a5h, 081h, 0bdh, 099h, 081h, 07eh
00014D19 81A581BD99817E      <1>
00014D20 7EFFDBFFC3E7FF7E6C- <1>     db  07eh, 0ffh, 0dbh, 0ffh, 0c3h, 0e7h, 0ffh, 07eh, 06ch, 0feh, 0feh, 0feh, 07ch, 038h, 010h, 000h
00014D29 FEFEFE7C381000      <1>
00014D30 10387CFE7C38100038- <1>     db  010h, 038h, 07ch, 0feh, 07ch, 038h, 010h, 000h, 038h, 07ch, 038h, 0feh, 0feh, 07ch, 038h, 07ch
00014D39 7C38FEFE7C387C      <1>
00014D40 1010387CFE7C387C00- <1>     db  010h, 010h, 038h, 07ch, 0feh, 07ch, 038h, 07ch, 000h, 000h, 018h, 03ch, 03ch, 018h, 000h, 000h
00014D49 00183C3C180000      <1>
00014D50 FFFFE7C3C3E7FFFF00- <1>     db  0ffh, 0ffh, 0e7h, 0c3h, 0c3h, 0e7h, 0ffh, 0ffh, 000h, 03ch, 066h, 042h, 042h, 066h, 03ch, 000h
00014D59 3C664242663C00      <1>
00014D60 FFC399BDBD99C3FF0F- <1>     db  0ffh, 0c3h, 099h, 0bdh, 0bdh, 099h, 0c3h, 0ffh, 00fh, 007h, 00fh, 07dh, 0cch, 0cch, 0cch, 078h
00014D69 070F7DCCCCCC78      <1>
00014D70 3C6666663C187E183F- <1>     db  03ch, 066h, 066h, 066h, 03ch, 018h, 07eh, 018h, 03fh, 033h, 03fh, 030h, 030h, 070h, 0f0h, 0e0h
00014D79 333F303070F0E0      <1>
00014D80 7F637F636367E6C099- <1>     db  07fh, 063h, 07fh, 063h, 063h, 067h, 0e6h, 0c0h, 099h, 05ah, 03ch, 0e7h, 0e7h, 03ch, 05ah, 099h
00014D89 5A3CE7E73C5A99      <1>
00014D90 80E0F8FEF8E0800002- <1>     db  080h, 0e0h, 0f8h, 0feh, 0f8h, 0e0h, 080h, 000h, 002h, 00eh, 03eh, 0feh, 03eh, 00eh, 002h, 000h
00014D99 0E3EFE3E0E0200      <1>
00014DA0 183C7E18187E3C1866- <1>     db  018h, 03ch, 07eh, 018h, 018h, 07eh, 03ch, 018h, 066h, 066h, 066h, 066h, 066h, 000h, 066h, 000h
00014DA9 66666666006600      <1>
00014DB0 7FDBDB7B1B1B1B003E- <1>     db  07fh, 0dbh, 0dbh, 07bh, 01bh, 01bh, 01bh, 000h, 03eh, 063h, 038h, 06ch, 06ch, 038h, 0cch, 078h
00014DB9 63386C6C38CC78      <1>
00014DC0 000000007E7E7E0018- <1>     db  000h, 000h, 000h, 000h, 07eh, 07eh, 07eh, 000h, 018h, 03ch, 07eh, 018h, 07eh, 03ch, 018h, 0ffh
00014DC9 3C7E187E3C18FF      <1>
00014DD0 183C7E181818180018- <1>     db  018h, 03ch, 07eh, 018h, 018h, 018h, 018h, 000h, 018h, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h
00014DD9 1818187E3C1800      <1>
00014DE0 00180CFE0C18000000- <1>     db  000h, 018h, 00ch, 0feh, 00ch, 018h, 000h, 000h, 000h, 030h, 060h, 0feh, 060h, 030h, 000h, 000h
00014DE9 3060FE60300000      <1>
00014DF0 0000C0C0C0FE000000- <1>     db  000h, 000h, 0c0h, 0c0h, 0c0h, 0feh, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h
00014DF9 2466FF66240000      <1>
00014E00 00183C7EFFFF000000- <1>     db  000h, 018h, 03ch, 07eh, 0ffh, 0ffh, 000h, 000h, 000h, 0ffh, 0ffh, 07eh, 03ch, 018h, 000h, 000h
00014E09 FFFF7E3C180000      <1>
00014E10 0000000000000000030- <1>    db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 030h, 078h, 078h, 030h, 030h, 000h, 030h, 000h
00014E19 78783030003000      <1>
00014E20 6C6C6C00000000006C- <1>     db  06ch, 06ch, 06ch, 000h, 000h, 000h, 000h, 000h, 06ch, 06ch, 0feh, 06ch, 0feh, 06ch, 06ch, 000h
00014E29 6CFE6CFE6C6C00      <1>
00014E30 307CC0780CF8300000- <1>     db  030h, 07ch, 0c0h, 078h, 00ch, 0f8h, 030h, 000h, 000h, 0c6h, 0cch, 018h, 030h, 066h, 0c6h, 000h
00014E39 C6CC183066C600      <1>
00014E40 386C3876DCCC760060- <1>     db  038h, 06ch, 038h, 076h, 0dch, 0cch, 076h, 000h, 060h, 060h, 0c0h, 000h, 000h, 000h, 000h, 000h
00014E49 60C00000000000      <1>
00014E50 1830606060300180060- <1>    db  018h, 030h, 060h, 060h, 060h, 030h, 018h, 000h, 060h, 030h, 018h, 018h, 018h, 030h, 060h, 000h
00014E59 30181818306000      <1>
00014E60 00663CFF3C66000000- <1>     db  000h, 066h, 03ch, 0ffh, 03ch, 066h, 000h, 000h, 000h, 030h, 030h, 0fch, 030h, 030h, 000h, 000h
00014E69 3030FC30300000      <1>
00014E70 0000000030306000- <1>       db  000h, 000h, 000h, 000h, 000h, 030h, 030h, 060h, 000h, 000h, 000h, 0fch, 000h, 000h, 000h, 000h
00014E79 0000FC00000000      <1>
00014E80 0000000030300006- <1>       db  000h, 000h, 000h, 000h, 000h, 030h, 030h, 000h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 080h, 000h
00014E89 0C183060C08000      <1>
00014E90 7CC6CEDEF6E67C0030- <1>     db  07ch, 0c6h, 0ceh, 0deh, 0f6h, 0e6h, 07ch, 000h, 030h, 070h, 030h, 030h, 030h, 030h, 0fch, 000h
00014E99 7030303030FC00      <1>
00014EA0 78CC0C3860CCFC0078- <1>     db  078h, 0cch, 00ch, 038h, 060h, 0cch, 0fch, 000h, 078h, 0cch, 00ch, 038h, 00ch, 0cch, 078h, 000h
00014EA9 CC0C380CCC7800      <1>
00014EB0 1C3C6CCCFE0C1E00FC- <1>     db  01ch, 03ch, 06ch, 0cch, 0feh, 00ch, 01eh, 000h, 0fch, 0c0h, 0f8h, 00ch, 00ch, 0cch, 078h, 000h
00014EB9 C0F80C0CCC7800      <1>
00014EC0 3860C0F8CCCC7800FC- <1>     db  038h, 060h, 0c0h, 0f8h, 0cch, 0cch, 078h, 000h, 0fch, 0cch, 00ch, 018h, 030h, 030h, 030h, 000h
00014EC9 CC0C1830303000      <1>
00014ED0 78CCCC78CCCC780078- <1>     db  078h, 0cch, 0cch, 078h, 0cch, 0cch, 078h, 000h, 078h, 0cch, 0cch, 07ch, 00ch, 018h, 070h, 000h
00014ED9 CCCC7C0C187000      <1>
00014EE0 0030300000303000000- <1>    db  000h, 030h, 030h, 000h, 000h, 030h, 030h, 000h, 000h, 030h, 030h, 000h, 000h, 030h, 030h, 060h
00014EE9 30300000303060      <1>
00014EF0 183060C06030180000- <1>     db  018h, 030h, 060h, 0c0h, 060h, 030h, 018h, 000h, 000h, 000h, 0fch, 000h, 000h, 0fch, 000h, 000h
00014EF9 00FC0000FC0000      <1>
00014F00 6030180C1830600078- <1>     db  060h, 030h, 018h, 00ch, 018h, 030h, 060h, 000h, 078h, 0cch, 00ch, 018h, 030h, 000h, 030h, 000h
00014F09 CC0C1830003000      <1>
00014F10 7CC6DEDEDEC0780030- <1>     db  07ch, 0c6h, 0deh, 0deh, 0deh, 0c0h, 078h, 000h, 030h, 078h, 0cch, 0cch, 0fch, 0cch, 0cch, 000h
00014F19 78CCCCFCCCCC00      <1>
00014F20 FC66667C6666FC003C- <1>     db  0fch, 066h, 066h, 07ch, 066h, 066h, 0fch, 000h, 03ch, 066h, 0c0h, 0c0h, 0c0h, 066h, 03ch, 000h
00014F29 66C0C0C0663C00      <1>
00014F30 F86C6666666CF800FE- <1>     db  0f8h, 06ch, 066h, 066h, 066h, 06ch, 0f8h, 000h, 0feh, 062h, 068h, 078h, 068h, 062h, 0feh, 000h
00014F39 6268786862FE00      <1>
00014F40 FE6268786860F0003C- <1>     db  0feh, 062h, 068h, 078h, 068h, 060h, 0f0h, 000h, 03ch, 066h, 0c0h, 0c0h, 0ceh, 066h, 03eh, 000h
00014F49 66C0C0CE663E00      <1>
00014F50 CCCCCCFCCCCCCC0078- <1>     db  0cch, 0cch, 0cch, 0fch, 0cch, 0cch, 0cch, 000h, 078h, 030h, 030h, 030h, 030h, 030h, 078h, 000h
00014F59 30303030307800      <1>
00014F60 1E0C0C0CCCCC7800E6- <1>     db  01eh, 00ch, 00ch, 00ch, 0cch, 0cch, 078h, 000h, 0e6h, 066h, 06ch, 078h, 06ch, 066h, 0e6h, 000h
00014F69 666C786C66E600      <1>
00014F70 F06060606266FE00C6- <1>     db  0f0h, 060h, 060h, 060h, 062h, 066h, 0feh, 000h, 0c6h, 0eeh, 0feh, 0feh, 0d6h, 0c6h, 0c6h, 000h
00014F79 EEFEFED6C6C600      <1>
00014F80 C6E6F6DECEC6C60038- <1>     db  0c6h, 0e6h, 0f6h, 0deh, 0ceh, 0c6h, 0c6h, 000h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 06ch, 038h, 000h
00014F89 6CC6C6C66C3800      <1>
00014F90 FC66667C6060F00078- <1>     db  0fch, 066h, 066h, 07ch, 060h, 060h, 0f0h, 000h, 078h, 0cch, 0cch, 0cch, 0dch, 078h, 01ch, 000h
00014F99 CCCCCCDC781C00      <1>
00014FA0 FC66667C6C66E60078- <1>     db  0fch, 066h, 066h, 07ch, 06ch, 066h, 0e6h, 000h, 078h, 0cch, 0e0h, 070h, 01ch, 0cch, 078h, 000h
00014FA9 CCE0701CCC7800      <1>
00014FB0 FCB4303030307800CC- <1>     db  0fch, 0b4h, 030h, 030h, 030h, 030h, 078h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 0fch, 000h
00014FB9 CCCCCCCCCCFC00      <1>
00014FC0 CCCCCCCCCC783000C6- <1>     db  0cch, 0cch, 0cch, 0cch, 0cch, 078h, 030h, 000h, 0c6h, 0c6h, 0c6h, 0d6h, 0feh, 0eeh, 0c6h, 000h
00014FC9 C6C6D6FEEEC600      <1>
00014FD0 C6C66C38386CC600CC- <1>     db  0c6h, 0c6h, 06ch, 038h, 038h, 06ch, 0c6h, 000h, 0cch, 0cch, 0cch, 078h, 030h, 030h, 078h, 000h
00014FD9 CCCC7830307800      <1>
00014FE0 FEC68C183266FE0078- <1>     db  0feh, 0c6h, 08ch, 018h, 032h, 066h, 0feh, 000h, 078h, 060h, 060h, 060h, 060h, 060h, 078h, 000h
00014FE9 60606060607800      <1>
00014FF0 C06030180C06020078- <1>     db  0c0h, 060h, 030h, 018h, 00ch, 006h, 002h, 000h, 078h, 018h, 018h, 018h, 018h, 018h, 078h, 000h
00014FF9 18181818187800      <1>
00015000 10386CC600000000000- <1>    db  010h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh
00015009 000000000000FF      <1>
```

```
173 00015010 303018000000000000- <1>     db  030h, 030h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 078h, 00ch, 07ch, 0cch, 076h, 000h
173 00015019 00780C7CCC7600      <1>
174 00015020 E060607C6666DC0000- <1>     db  0e0h, 060h, 060h, 07ch, 066h, 066h, 0dch, 000h, 000h, 000h, 078h, 0cch, 0c0h, 0cch, 078h, 000h
174 00015029 0078CCC0CC7800      <1>
175 00015030 1C0C0C7CCCCC760000- <1>     db  01ch, 00ch, 00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h
175 00015039 0078CCFCC07800      <1>
176 00015040 386C60F06060F00000- <1>     db  038h, 06ch, 060h, 0f0h, 060h, 060h, 0f0h, 000h, 000h, 000h, 076h, 0cch, 0cch, 07ch, 00ch, 0f8h
176 00015049 0076CCCC7C0CF8      <1>
177 00015050 E0606C766666E60030- <1>     db  0e0h, 060h, 06ch, 076h, 066h, 066h, 0e6h, 000h, 030h, 000h, 070h, 030h, 030h, 030h, 078h, 000h
177 00015059 00703030307800      <1>
178 00015060 0C000C0C0CCCCC78E0- <1>     db  00ch, 000h, 00ch, 00ch, 00ch, 0cch, 0cch, 078h, 0e0h, 060h, 066h, 06ch, 078h, 06ch, 0e6h, 000h
178 00015069 60666C786CE600      <1>
179 00015070 703030303030780000- <1>     db  070h, 030h, 030h, 030h, 030h, 030h, 078h, 000h, 000h, 000h, 0cch, 0feh, 0feh, 0d6h, 0c6h, 000h
179 00015079 00CCFEFED6C600      <1>
180 00015080 0000F8CCCCCCCC0000- <1>     db  000h, 000h, 0f8h, 0cch, 0cch, 0cch, 0cch, 000h, 000h, 000h, 078h, 0cch, 0cch, 0cch, 078h, 000h
180 00015089 0078CCCCCC7800      <1>
181 00015090 0000DC66667C60F000- <1>     db  000h, 000h, 0dch, 066h, 066h, 07ch, 060h, 0f0h, 000h, 000h, 076h, 0cch, 0cch, 07ch, 00ch, 01eh
181 00015099 0076CCCC7C0C1E      <1>
182 000150A0 0000DC766660F00000- <1>     db  000h, 000h, 0dch, 076h, 066h, 060h, 0f0h, 000h, 000h, 000h, 07ch, 0c0h, 078h, 00ch, 0f8h, 000h
182 000150A9 007CC0780CF800      <1>
183 000150B0 10307C303034180000- <1>     db  010h, 030h, 07ch, 030h, 030h, 034h, 018h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 076h, 000h
183 000150B9 00CCCCCCCC7600      <1>
184 000150C0 0000CCCCCC78300000- <1>     db  000h, 000h, 0cch, 0cch, 0cch, 078h, 030h, 000h, 000h, 000h, 0c6h, 0d6h, 0feh, 0feh, 06ch, 000h
184 000150C9 00C6D6FEFE6C00      <1>
185 000150D0 0000C66C386CC60000- <1>     db  000h, 000h, 0c6h, 06ch, 038h, 06ch, 0c6h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 07ch, 00ch, 0f8h
185 000150D9 00CCCCCC7C0CF8      <1>
186 000150E0 0000FC983064FC001C- <1>     db  000h, 000h, 0fch, 098h, 030h, 064h, 0fch, 000h, 01ch, 030h, 030h, 0e0h, 030h, 030h, 01ch, 000h
186 000150E9 3030E030301C00      <1>
187 000150F0 18181800181818 00E0- <1>     db  018h, 018h, 018h, 000h, 018h, 018h, 018h, 000h, 0e0h, 030h, 030h, 01ch, 030h, 030h, 0e0h, 000h
187 000150F9 30301C3030E000      <1>
188 00015100 76DC00000000000000- <1>     db  076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 000h
188 00015109 10386CC6C6FE00      <1>
189 00015110 78CCC0CC78180C7800- <1>     db  078h, 0cch, 0c0h, 0cch, 078h, 018h, 00ch, 078h, 000h, 0cch, 000h, 0cch, 0cch, 0cch, 07eh, 000h
189 00015119 CC00CCCCCC7E00      <1>
190 00015120 1C0078CCFCC078007E- <1>     db  01ch, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h, 07eh, 0c3h, 03ch, 006h, 03eh, 066h, 03fh, 000h
190 00015129 C33C063E663F00      <1>
191 00015130 CC00780C7CCC7E00E0- <1>     db  0cch, 000h, 078h, 00ch, 07ch, 0cch, 07eh, 000h, 0e0h, 000h, 078h, 00ch, 07ch, 0cch, 07eh, 000h
191 00015139 00780C7CCC7E00      <1>
192 00015140 3030780C7CCC7E0000- <1>     db  030h, 030h, 078h, 00ch, 07ch, 0cch, 07eh, 000h, 000h, 000h, 078h, 0c0h, 0c0h, 078h, 00ch, 038h
192 00015149 0078C0C0780C38      <1>
193 00015150 7EC33C667E603C00CC- <1>     db  07eh, 0c3h, 03ch, 066h, 07eh, 060h, 03ch, 000h, 0cch, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h
193 00015159 0078CCFCC07800      <1>
194 00015160 E00078CCFCC07800CC- <1>     db  0e0h, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h, 0cch, 000h, 070h, 030h, 030h, 030h, 078h, 000h
194 00015169 00703030307800      <1>
195 00015170 7CC6381818183C00E0- <1>     db  07ch, 0c6h, 038h, 018h, 018h, 018h, 03ch, 000h, 0e0h, 000h, 070h, 030h, 030h, 030h, 078h, 000h
195 00015179 00703030307800      <1>
196 00015180 C6386CC6FEC6C60030- <1>     db  0c6h, 038h, 06ch, 0c6h, 0feh, 0c6h, 0c6h, 000h, 030h, 030h, 000h, 078h, 0cch, 0fch, 0cch, 000h
196 00015189 300078CCFCCC00      <1>
197 00015190 1C00FC607860FC0000- <1>     db  01ch, 000h, 0fch, 060h, 078h, 060h, 0fch, 000h, 000h, 000h, 07fh, 00ch, 07fh, 0cch, 07fh, 000h
197 00015199 007F0C7FCC7F00      <1>
198 000151A0 3E6CCCFECCCCCE0078- <1>     db  03eh, 06ch, 0cch, 0feh, 0cch, 0cch, 0ceh, 000h, 078h, 0cch, 000h, 078h, 0cch, 0cch, 078h, 000h
198 000151A9 CC0078CCCC7800      <1>
199 000151B0 00CC0078CCCC780000- <1>     db  000h, 0cch, 000h, 078h, 0cch, 0cch, 078h, 000h, 000h, 0e0h, 000h, 078h, 0cch, 0cch, 078h, 000h
199 000151B9 E00078CCCC7800      <1>
200 000151C0 78CC00CCCCCC7E0000- <1>     db  078h, 0cch, 000h, 0cch, 0cch, 0cch, 07eh, 000h, 000h, 0e0h, 000h, 0cch, 0cch, 0cch, 07eh, 000h
200 000151C9 E000CCCCCC7E00      <1>
201 000151D0 00CC00CCCC7C0CF8C3- <1>     db  000h, 0cch, 000h, 0cch, 0cch, 07ch, 00ch, 0f8h, 0c3h, 018h, 03ch, 066h, 066h, 03ch, 018h, 000h
201 000151D9 183C66663C1800      <1>
202 000151E0 CC00CCCCCCCC780018- <1>     db  0cch, 000h, 0cch, 0cch, 0cch, 0cch, 078h, 000h, 018h, 018h, 07eh, 0c0h, 0c0h, 07eh, 018h, 018h
202 000151E9 187EC0C07E1818      <1>
203 000151F0 386C64F060E6FC00CC- <1>     db  038h, 06ch, 064h, 0f0h, 060h, 0e6h, 0fch, 000h, 0cch, 0cch, 078h, 0fch, 030h, 0fch, 030h, 030h
203 000151F9 CC78FC30FC3030      <1>
204 00015200 F8CCCCFAC6CFC6C70E- <1>     db  0f8h, 0cch, 0cch, 0fah, 0c6h, 0cfh, 0c6h, 0c7h, 00eh, 01bh, 018h, 03ch, 018h, 018h, 0d8h, 070h
204 00015209 1B183C1818D870      <1>
205 00015210 1C00780C7CCC7E0038- <1>     db  01ch, 000h, 078h, 00ch, 07ch, 0cch, 07eh, 000h, 038h, 000h, 070h, 030h, 030h, 030h, 078h, 000h
205 00015219 00703030307800      <1>
206 00015220 001C0078CCCC780000- <1>     db  000h, 01ch, 000h, 078h, 0cch, 0cch, 078h, 000h, 000h, 01ch, 000h, 0cch, 0cch, 0cch, 07eh, 000h
206 00015229 1C00CCCCCC7E00      <1>
207 00015230 00F800F8CCCCCC00FC- <1>     db  000h, 0f8h, 000h, 0f8h, 0cch, 0cch, 0cch, 000h, 0fch, 000h, 0cch, 0ech, 0fch, 0dch, 0cch, 000h
207 00015239 00CCECFCDCCC00      <1>
208 00015240 3C6C6C3E007E000038- <1>     db  03ch, 06ch, 06ch, 03eh, 000h, 07eh, 000h, 000h, 038h, 06ch, 06ch, 038h, 000h, 07ch, 000h, 000h
208 00015249 6C6C38007C0000      <1>
209 00015250 30003060C0CC780000- <1>     db  030h, 000h, 030h, 060h, 0c0h, 0cch, 078h, 000h, 000h, 000h, 000h, 0fch, 0c0h, 0c0h, 000h, 000h
209 00015259 0000FCC0C00000      <1>
210 00015260 000000FC0C0C0000C3- <1>     db  000h, 000h, 000h, 0fch, 00ch, 00ch, 000h, 000h, 0c3h, 0c6h, 0cch, 0deh, 033h, 066h, 0cch, 00fh
210 00015269 C6CCDE3366CC0F      <1>
211 00015270 C3C6CCDB376FCF0318- <1>     db  0c3h, 0c6h, 0cch, 0dbh, 037h, 06fh, 0cfh, 003h, 018h, 018h, 000h, 018h, 018h, 018h, 018h, 000h
211 00015279 18001818181800      <1>
212 00015280 003366CC6633000000- <1>     db  000h, 033h, 066h, 0cch, 066h, 033h, 000h, 000h, 000h, 0cch, 066h, 033h, 066h, 0cch, 000h, 000h
212 00015289 CC663366CC0000      <1>
213 00015290 228822882288228855- <1>     db  022h, 088h, 022h, 088h, 022h, 088h, 022h, 088h, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah
213 00015299 AA55AA55AA55AA      <1>
214 000152A0 DB77DBEEDB77DBEE18- <1>     db  0dbh, 077h, 0dbh, 0eeh, 0dbh, 077h, 0dbh, 0eeh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
214 000152A9 18181818181818      <1>
215 000152B0 18181818F818181818- <1>     db  018h, 018h, 018h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 018h, 018h, 018h
215 000152B9 18F818F8181818      <1>
216 000152C0 36363636F636363600- <1>     db  036h, 036h, 036h, 036h, 0f6h, 036h, 036h, 036h, 000h, 000h, 000h, 000h, 0feh, 036h, 036h, 036h
216 000152C9 000000FE363636      <1>
217 000152D0 0000F818F818181836- <1>     db  000h, 000h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 036h, 036h, 0f6h, 006h, 0f6h, 036h, 036h, 036h
217 000152D9 36F606F6363636      <1>
218 000152E0 3636363636363600- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 000h, 000h, 0feh, 006h, 0f6h, 036h, 036h, 036h
218 000152E9 00FE06F6363636      <1>
219 000152F0 3636F606FE00000036- <1>     db  036h, 036h, 0f6h, 006h, 0feh, 000h, 000h, 000h, 036h, 036h, 036h, 036h, 0feh, 000h, 000h, 000h
219 000152F9 363636FE000000      <1>
220 00015300 1818F818F800000000- <1>     db  018h, 018h, 0f8h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 018h, 018h
220 00015309 000000F8181818      <1>
221 00015310 181818181F00000018- <1>     db  018h, 018h, 018h, 018h, 01fh, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 0ffh, 000h, 000h, 000h
221 00015319 181818FF000000      <1>
222 00015320 00000000FF18181818- <1>     db  000h, 000h, 000h, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 018h, 018h
222 00015329 1818181F181818      <1>
223 00015330 00000000FF00000018- <1>     db  000h, 000h, 000h, 000h, 0ffh, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h
223 00015339 181818FF181818      <1>
224 00015340 18181F181F18181836- <1>     db  018h, 018h, 01fh, 018h, 01fh, 018h, 018h, 018h, 036h, 036h, 036h, 036h, 037h, 036h, 036h, 036h
224 00015349 36363637363636      <1>
225 00015350 363637303F00000000- <1>     db  036h, 036h, 037h, 030h, 03fh, 000h, 000h, 000h, 000h, 000h, 03fh, 030h, 037h, 036h, 036h, 036h
225 00015359 003F3037363636      <1>
226 00015360 3636F700FF00000000- <1>     db  036h, 036h, 0f7h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0f7h, 036h, 036h, 036h
226 00015369 00FF00F7363636      <1>
227 00015370 3636373037363600- <1>     db  036h, 036h, 037h, 030h, 037h, 036h, 036h, 036h, 000h, 000h, 0ffh, 000h, 0ffh, 000h, 000h, 000h
227 00015379 00FF00FF000000      <1>
228 00015380 3636F700F736363618- <1>     db  036h, 036h, 0f7h, 000h, 0f7h, 036h, 036h, 036h, 018h, 018h, 0ffh, 000h, 0ffh, 000h, 000h, 000h
228 00015389 18FF00FF000000      <1>
229 00015390 36363636FF00000000- <1>     db  036h, 036h, 036h, 036h, 0ffh, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 018h, 018h, 018h
229 00015399 00FF00FF181818      <1>
230 000153A0 00000000FF36363636- <1>     db  000h, 000h, 000h, 000h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 03fh, 000h, 000h, 000h
230 000153A9 3636363F000000      <1>
231 000153B0 18181F181F00000000- <1>     db  018h, 018h, 01fh, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 01fh, 018h, 01fh, 018h, 018h, 018h
231 000153B9 001F181F181818      <1>
232 000153C0 000000003F36363636- <1>     db  000h, 000h, 000h, 000h, 03fh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 036h, 036h, 036h
232 000153C9 363636FF363636      <1>
233 000153D0 1818FF18FF18181818- <1>     db  018h, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 000h, 000h, 000h
233 000153D9 181818F8000000      <1>
234 000153E0 000000001F181818FF- <1>     db  000h, 000h, 000h, 000h, 01fh, 018h, 018h, 018h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
234 000153E9 FFFFFFFFFFFFFF      <1>
```

```
235 000153F0 00000000FFFFFFFFF0- <1>     db  000h, 000h, 000h, 000h, 0ffh, 0ffh, 0ffh, 0ffh, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h
235 000153F9 F0F0F0F0F0F0F0      <1>
236 00015400 0F0F0F0F0F0F0FFF- <1>     db  00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h
236 00015409 FFFFFF00000000      <1>
237 00015410 000076DCC8DC760000- <1>     db  000h, 000h, 076h, 0dch, 0c8h, 0dch, 076h, 000h, 000h, 078h, 0cch, 0f8h, 0cch, 0f8h, 0c0h, 0c0h
237 00015419 78CCF8CCF8C0C0      <1>
238 00015420 00FCCCC0C0C0C00000- <1>     db  000h, 0fch, 0cch, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 0feh, 06ch, 06ch, 06ch, 06ch, 06ch, 000h
238 00015429 FE6C6C6C6C6C00      <1>
239 00015430 FCCC603060CCFC0000- <1>     db  0fch, 0cch, 060h, 030h, 060h, 0cch, 0fch, 000h, 000h, 000h, 07eh, 0d8h, 0d8h, 0d8h, 070h, 000h
239 00015439 007ED8D8D87000      <1>
240 00015440 00666666667C60C000- <1>     db  000h, 066h, 066h, 066h, 066h, 07ch, 060h, 0c0h, 000h, 076h, 0dch, 018h, 018h, 018h, 018h, 000h
240 00015449 76DC1818181800      <1>
241 00015450 FC3078CCCC7830FC38- <1>     db  0fch, 030h, 078h, 0cch, 0cch, 078h, 030h, 0fch, 038h, 06ch, 0c6h, 0feh, 0c6h, 06ch, 038h, 000h
241 00015459 6CC6FEC66C3800      <1>
242 00015460 386CC6C66C6CEE001C- <1>     db  038h, 06ch, 0c6h, 0c6h, 06ch, 06ch, 0eeh, 000h, 01ch, 030h, 018h, 07ch, 0cch, 0cch, 078h, 000h
242 00015469 30187CCCCC7800      <1>
243 00015470 00007EDBDB7E000006- <1>     db  000h, 000h, 07eh, 0dbh, 0dbh, 07eh, 000h, 000h, 006h, 00ch, 07eh, 0dbh, 0dbh, 07eh, 060h, 0c0h
243 00015479 0C7EDBDB7E60C0      <1>
244 00015480 3860C0F8C060380078- <1>     db  038h, 060h, 0c0h, 0f8h, 0c0h, 060h, 038h, 000h, 078h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 000h
244 00015489 CCCCCCCCCCCC00      <1>
245 00015490 00FC00FC00FC000030- <1>     db  000h, 0fch, 000h, 0fch, 000h, 0fch, 000h, 000h, 030h, 030h, 0fch, 030h, 030h, 000h, 0fch, 000h
245 00015499 30FC303000FC00      <1>
246 000154A0 603018306000FC0018- <1>     db  060h, 030h, 018h, 030h, 060h, 000h, 0fch, 000h, 018h, 030h, 060h, 030h, 018h, 000h, 0fch, 000h
246 000154A9 3060301800FC00      <1>
247 000154B0 0E1B1B181818181818- <1>     db  00eh, 01bh, 01bh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0d8h, 0d8h, 070h
247 000154B9 18181818D8D870      <1>
248 000154C0 303000FC0030300000- <1>     db  030h, 030h, 000h, 0fch, 000h, 030h, 030h, 000h, 000h, 076h, 0dch, 000h, 076h, 0dch, 000h, 000h
248 000154C9 76DC0076DC0000      <1>
249 000154D0 386C6C380000000000- <1>     db  038h, 06ch, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h
249 000154D9 00001818000000      <1>
250 000154E0 000000001800000000F- <1>     db  000h, 000h, 000h, 000h, 018h, 000h, 000h, 000h, 00fh, 00ch, 00ch, 00ch, 0ech, 06ch, 03ch, 01ch
250 000154E9 0C0C0CEC6C3C1C      <1>
251 000154F0 786C6C6C6C00000070- <1>     db  078h, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 070h, 018h, 030h, 060h, 078h, 000h, 000h, 000h
251 000154F9 18306078000000      <1>
252 00015500 00003C3C3C3C000000- <1>     db  000h, 000h, 03ch, 03ch, 03ch, 03ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
252 00015509 00000000000000      <1>
253                              <1> vgafont14:
254 00015510 00000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
254 00015519 00000000000000      <1>
255 00015520 7E81A58181BD99817E- <1>     db  07eh, 081h, 0a5h, 081h, 081h, 0bdh, 099h, 081h, 07eh, 000h, 000h, 000h, 000h, 07eh, 0ffh
255 00015529 00000000007EFF      <1>
256 00015530 DBFFFFC3E7FF7E0000- <1>     db  0dbh, 0ffh, 0ffh, 0c3h, 0e7h, 0ffh, 07eh, 000h, 000h, 000h, 000h, 000h, 000h, 06ch, 0feh, 0feh
256 00015539 000000006CFEFE      <1>
257 00015540 FEFE7C3810000000000- <1>     db  0feh, 0feh, 07ch, 038h, 010h, 000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 07ch, 0feh, 07ch
257 00015549 000010387CFE7C      <1>
258 00015550 3810000000000000018- <1>     db  038h, 010h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 03ch, 0e7h, 0e7h, 0e7h, 018h, 018h
258 00015559 3C3CE7E7E71818      <1>
259 00015560 3C0000000000183C7E- <1>     db  03ch, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 07eh, 0ffh, 0ffh, 07eh, 018h, 018h, 03ch, 000h
259 00015569 FFFF7E18183C00      <1>
260 00015570 00000000000000183C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 03ch, 018h, 000h, 000h, 000h, 000h, 000h
260 00015579 3C180000000000      <1>
261 00015580 FFFFFFFFFE7C3C3E7- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0e7h, 0c3h, 0c3h, 0e7h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h
261 00015589 FFFFFFFFFF0000      <1>
262 00015590 00003C664242663C00- <1>     db  000h, 000h, 03ch, 066h, 042h, 042h, 066h, 03ch, 000h, 000h, 000h, 000h, 0ffh, 0ffh, 0ffh, 0ffh
262 00015599 000000FFFFFFFF      <1>
263 000155A0 C399BDBD99C3FFFFFF- <1>     db  0c3h, 099h, 0bdh, 0bdh, 099h, 0c3h, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 01eh, 00eh, 01ah, 032h
263 000155A9 FF00001E0E1A32      <1>
264 000155B0 78CCCCCC7800000000- <1>     db  078h, 0cch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h, 000h, 03ch, 066h, 066h, 066h, 03ch, 018h
264 000155B9 003C6666663C18      <1>
265 000155C0 7E1818000000000003F- <1>     db  07eh, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 03fh, 033h, 03fh, 030h, 030h, 030h, 070h, 0f0h
265 000155C9 333F30303070F0      <1>
266 000155D0 E000000000007F637F- <1>     db  0e0h, 000h, 000h, 000h, 000h, 000h, 07fh, 063h, 07fh, 063h, 063h, 063h, 067h, 0e7h, 0e6h, 0c0h
266 000155D9 63636367E7E6C0      <1>
267 000155E0 000000001818DB3CE7- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 0dbh, 03ch, 0e7h, 03ch, 0dbh, 018h, 018h, 000h, 000h, 000h
267 000155E9 3CDB1818000000      <1>
268 000155F0 000080C0E0F8FEF8E0- <1>     db  000h, 000h, 080h, 0c0h, 0e0h, 0f8h, 0feh, 0f8h, 0e0h, 0c0h, 080h, 000h, 000h, 000h, 000h, 000h
268 000155F9 C0800000000000      <1>
269 00015600 02060E3EFE3E0E0602- <1>     db  002h, 006h, 00eh, 03eh, 0feh, 03eh, 00eh, 006h, 002h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch
269 00015609 0000000000183C      <1>
270 00015610 7E1818187E3C180000- <1>     db  07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h
270 00015619 00000066666666      <1>
271 00015620 666600666600000000- <1>     db  066h, 066h, 000h, 066h, 066h, 000h, 000h, 000h, 000h, 000h, 07fh, 0dbh, 0dbh, 0dbh, 07bh, 01bh
271 00015629 007FDBDBDB7B1B      <1>
272 00015630 1B1B1B000000007CC6- <1>     db  01bh, 01bh, 01bh, 000h, 000h, 000h, 000h, 07ch, 0c6h, 060h, 038h, 06ch, 0c6h, 0c6h, 06ch, 038h
272 00015639 60386CC6C66C38      <1>
273 00015640 0CC67C000000000000- <1>     db  00ch, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 0feh, 0feh, 000h
273 00015649 000000FEFEFE00      <1>
274 00015650 00000000183C7E1818- <1>     db  000h, 000h, 000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 07eh, 000h, 000h
274 00015659 187E3C187E0000      <1>
275 00015660 0000183C7E18181818- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h
275 00015669 18180000000000      <1>
276 00015670 1818181818187E3C18- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
276 00015679 00000000000000      <1>
277 00015680 180CFE0C1800000000- <1>     db  018h, 00ch, 0feh, 00ch, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 030h, 060h
277 00015689 00000000003060      <1>
278 00015690 FE60300000000000000- <1>     db  0feh, 060h, 030h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0c0h, 0c0h, 0c0h
278 00015699 00000000C0C0C0      <1>
279 000156A0 FE0000000000000000- <1>     db  0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 028h, 06ch, 0feh, 06ch, 028h, 000h
279 000156A9 00286CFE6C2800      <1>
280 000156B0 00000000000000001038- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 038h, 07ch, 07ch, 0feh, 0feh, 000h, 000h
280 000156B9 387C7CFEFE0000      <1>
281 000156C0 0000000000FEFE7C7C- <1>     db  000h, 000h, 000h, 000h, 000h, 0feh, 0feh, 07ch, 07ch, 038h, 038h, 010h, 000h, 000h, 000h, 000h
281 000156C9 38381000000000      <1>
282 000156D0 00000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
282 000156D9 00000000000000      <1>
283 000156E0 183C3C3C1818001818- <1>     db  018h, 03ch, 03ch, 03ch, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 066h, 066h, 066h
283 000156E9 00000000666666      <1>
284 000156F0 240000000000000000- <1>     db  024h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 06ch, 06ch, 0feh, 06ch
284 000156F9 0000006C6CFE6C      <1>
285 00015700 6C6CFE6C6C00000018- <1>     db  06ch, 06ch, 0feh, 06ch, 06ch, 000h, 000h, 000h, 018h, 018h, 07ch, 0c6h, 0c2h, 0c0h, 07ch, 006h
285 00015709 187CC6C2C07C06      <1>
286 00015710 86C67C181800000000- <1>     db  086h, 0c6h, 07ch, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 0c2h, 0c6h, 00ch, 018h, 030h, 066h
286 00015719 00C2C60C183066      <1>
287 00015720 C60000000000386C6C- <1>     db  0c6h, 000h, 000h, 000h, 000h, 000h, 038h, 06ch, 06ch, 038h, 076h, 0dch, 0cch, 0cch, 076h, 000h
287 00015729 3876DCCCCC7600      <1>
288 00015730 000000303030600000- <1>     db  000h, 000h, 000h, 030h, 030h, 030h, 060h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
288 00015739 00000000000000      <1>
289 00015740 00000C183030303030- <1>     db  000h, 000h, 00ch, 018h, 030h, 030h, 030h, 030h, 030h, 018h, 00ch, 000h, 000h, 000h, 000h, 000h
289 00015749 180C0000000000      <1>
290 00015750 30180C0C0C0C0C1830- <1>     db  030h, 018h, 00ch, 00ch, 00ch, 00ch, 00ch, 018h, 030h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
290 00015759 00000000000000      <1>
291 00015760 663CFF3C6600000000- <1>     db  066h, 03ch, 0ffh, 03ch, 066h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h
291 00015769 00000000001818      <1>
292 00015770 7E1818000000000000- <1>     db  07eh, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
292 00015779 00000000000000      <1>
293 00015780 1818183000000000000- <1>     db  018h, 018h, 018h, 030h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 000h, 000h, 000h
293 00015789 000000FE000000      <1>
294 00015790 00000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h
294 00015799 00000000181800      <1>
295 000157A0 0000000002060C1830- <1>     db  000h, 000h, 000h, 000h, 002h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 080h, 000h, 000h, 000h, 000h
295 000157A9 60C08000000000      <1>
296 000157B0 00007CC6CEDEF6E6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0ceh, 0deh, 0f6h, 0e6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h
296 000157B9 C67C0000000000      <1>
297 000157C0 183878181818187E- <1>     db  018h, 038h, 078h, 018h, 018h, 018h, 018h, 018h, 07eh, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h
```

```
297 000157C9 00000000007CC6      <1>
298 000157D0 060C183060C6FE0000- <1>     db  006h, 00ch, 018h, 030h, 060h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 006h, 006h
298 000157D9 0000007CC60606      <1>
299 000157E0 3C0606C67C00000000- <1>     db  03ch, 006h, 006h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 00ch, 01ch, 03ch, 06ch, 0cch, 0feh
299 000157E9 000C1C3C6CCCFE      <1>
300 000157F0 0C0C1E0000000000FE- <1>     db  00ch, 00ch, 01eh, 000h, 000h, 000h, 000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 0fch, 006h, 006h, 0c6h
300 000157F9 C0C0C0FC0606C6      <1>
301 00015800 7C00000000003860C0- <1>     db  07ch, 000h, 000h, 000h, 000h, 000h, 038h, 060h, 0c0h, 0c0h, 0fch, 0c6h, 0c6h, 0c6h, 07ch, 000h
301 00015809 C0FCC6C6C67C00      <1>
302 00015810 00000000FEC6060C18- <1>     db  000h, 000h, 000h, 000h, 0feh, 0c6h, 006h, 00ch, 018h, 030h, 030h, 030h, 030h, 000h, 000h, 000h
302 00015819 30303030000000      <1>
303 00015820 00007CC6C6C67CC6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 07ch, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h
303 00015829 C67C00000000000     <1>
304 00015830 7CC6C6C67E06060C78- <1>     db  07ch, 0c6h, 0c6h, 0c6h, 07eh, 006h, 006h, 00ch, 078h, 000h, 000h, 000h, 000h, 000h, 000h, 018h
304 00015839 00000000000018      <1>
305 00015840 180000001818000000- <1>     db  018h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h
305 00015849 00000000181800      <1>
306 00015850 0000181830000000- <1>     db  000h, 000h, 018h, 018h, 030h, 000h, 000h, 000h, 000h, 000h, 006h, 00ch, 018h, 030h, 060h, 030h
306 00015859 00060C18306030      <1>
307 00015860 180C060000000000- <1>     db  018h, 00ch, 006h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 000h, 000h, 07eh, 000h
307 00015869 00007E00007E00      <1>
308 00015870 0000000000603018- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 00ch, 006h, 00ch, 018h, 030h, 060h, 000h
308 00015879 0C060C18306000      <1>
309 00015880 000000007CC6C60C18- <1>     db  000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 00ch, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h
309 00015889 18001818000000      <1>
310 00015890 00007CC6C6DEDEDEDC- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0deh, 0deh, 0deh, 0dch, 0c0h, 07ch, 000h, 000h, 000h, 000h, 000h
310 00015899 C07C0000000000      <1>
311 000158A0 10386CC6C6FEC6C6C6- <1>     db  010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h, 000h, 0fch, 066h
311 000158A9 0000000000FC66      <1>
312 000158B0 66667C666666FC0000- <1>     db  066h, 066h, 07ch, 066h, 066h, 066h, 0fch, 000h, 000h, 000h, 000h, 000h, 03ch, 066h, 0c2h, 0c0h
312 000158B9 0000003C66C2C0      <1>
313 000158C0 C0C0C2663C00000000- <1>     db  0c0h, 0c0h, 0c2h, 066h, 03ch, 000h, 000h, 000h, 000h, 000h, 0f8h, 06ch, 066h, 066h, 066h, 066h
313 000158C9 00F86C66666666      <1>
314 000158D0 666CF800000000000FE- <1>     db  066h, 06ch, 0f8h, 000h, 000h, 000h, 000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 062h, 066h
314 000158D9 66626878686266      <1>
315 000158E0 FE0000000000FE6662- <1>     db  0feh, 000h, 000h, 000h, 000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 060h, 060h, 0f0h, 000h
315 000158E9 6878686060F000      <1>
316 000158F0 000000003C66C2C0C0- <1>     db  000h, 000h, 000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0deh, 0c6h, 066h, 03ah, 000h, 000h, 000h
316 000158F9 DEC6663A000000      <1>
317 00015900 0000C6C6C6C6FEC6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h, 000h
317 00015909 C6C60000000000      <1>
318 00015910 3C1818181818183C- <1>     db  03ch, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 01eh, 00ch
318 00015919 00000000001E0C      <1>
319 00015920 0C0C0C0CCCCC780000- <1>     db  00ch, 00ch, 00ch, 00ch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h, 000h, 0e6h, 066h, 06ch, 06ch
319 00015929 000000E6666C6C      <1>
320 00015930 786C6C66E600000000- <1>     db  078h, 06ch, 06ch, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h, 0f0h, 060h, 060h, 060h, 060h, 060h
320 00015939 00F06060606060      <1>
321 00015940 6266FE0000000000C6- <1>     db  062h, 066h, 0feh, 000h, 000h, 000h, 000h, 000h, 0c6h, 0eeh, 0feh, 0feh, 0d6h, 0c6h, 0c6h, 0c6h
321 00015949 EEFEFED6C6C6C6      <1>
322 00015950 C60000000000C6E6F6- <1>     db  0c6h, 000h, 000h, 000h, 000h, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h, 0c6h, 0c6h, 000h
322 00015959 FEDECEC6C6C600      <1>
323 00015960 00000000386CC6C6C6- <1>     db  000h, 000h, 000h, 000h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h
323 00015969 C6C66C38000000      <1>
324 00015970 0000FC6666667C6060- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h, 000h
324 00015979 60F000000000000     <1>
325 00015980 7CC6C6C6C6D6DE7C0C- <1>     db  07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0d6h, 0deh, 07ch, 00ch, 00eh, 000h, 000h, 000h, 000h, 0fch, 066h
325 00015989 0E00000000FC66      <1>
326 00015990 66667C6C6666E60000- <1>     db  066h, 066h, 07ch, 06ch, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 060h
326 00015999 0000007CC6C660      <1>
327 000159A0 380CC6C67C00000000- <1>     db  038h, 00ch, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 07eh, 07eh, 05ah, 018h, 018h, 018h
327 000159A9 007E7E5A181818      <1>
328 000159B0 18183C0000000000C6- <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h
328 000159B9 C6C6C6C6C6C6C6      <1>
329 000159C0 7C0000000000C6C6C6- <1>     db  07ch, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 06ch, 038h, 010h, 000h
329 000159C9 C6C6C66C381000      <1>
330 000159D0 00000000C6C6C6C6D6- <1>     db  000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0d6h, 0d6h, 0feh, 07ch, 06ch, 000h, 000h, 000h
330 000159D9 D6FE7C6C000000      <1>
331 000159E0 0000C6C66C3838386C- <1>     db  000h, 000h, 0c6h, 0c6h, 06ch, 038h, 038h, 038h, 06ch, 0c6h, 0c6h, 000h, 000h, 000h, 000h, 000h
331 000159E9 C6C60000000000      <1>
332 000159F0 666666663C1818183C- <1>     db  066h, 066h, 066h, 066h, 03ch, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 0feh, 0c6h
332 000159F9 0000000000FEC6      <1>
333 00015A00 8C183060C2C6FE0000- <1>     db  08ch, 018h, 030h, 060h, 0c2h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h, 03ch, 030h, 030h, 030h
333 00015A09 0000003C303030      <1>
334 00015A10 303030303C00000000- <1>     db  030h, 030h, 030h, 030h, 03ch, 000h, 000h, 000h, 000h, 000h, 080h, 0c0h, 0e0h, 070h, 038h, 01ch
334 00015A19 0080C0E070381C      <1>
335 00015A20 0E060200000000003C- <1>     db  00eh, 006h, 002h, 000h, 000h, 000h, 000h, 000h, 03ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch
335 00015A29 0C0C0C0C0C0C0C      <1>
336 00015A30 3C000000010386CC600- <1>     db  03ch, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
336 00015A39 00000000000000      <1>
337 00015A40 0000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h
337 00015A49 0000000000FF00      <1>
338 00015A50 303018000000000000- <1>     db  030h, 030h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
338 00015A59 00000000000000      <1>
339 00015A60 000000780C7CCCCC76- <1>     db  000h, 000h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 0e0h, 060h
339 00015A69 0000000000E060      <1>
340 00015A70 60786C6666667C0000- <1>     db  060h, 078h, 06ch, 066h, 066h, 066h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch
340 00015A79 0000000000007C      <1>
341 00015A80 C6C0C0C67C00000000- <1>     db  0c6h, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 01ch, 00ch, 00ch, 03ch, 06ch, 0cch
341 00015A89 001C0C0C3C6CCC      <1>
342 00015A90 CCCC760000000000000- <1>     db  0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h
342 00015A99 00007CC6FEC0C6      <1>
343 00015AA0 7C0000000000386C64- <1>     db  07ch, 000h, 000h, 000h, 000h, 000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h, 060h, 0f0h, 000h
343 00015AA9 60F0606060F000      <1>
344 00015AB0 0000000000000076CC- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 07ch, 00ch, 0cch, 078h, 000h
344 00015AB9 CCCC7C0CCC7800      <1>
345 00015AC0 0000E060606C766666- <1>     db  000h, 000h, 0e0h, 060h, 060h, 06ch, 076h, 066h, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h
345 00015AC9 66E60000000000      <1>
346 00015AD0 181800381818183C- <1>     db  018h, 018h, 000h, 038h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 006h, 006h
346 00015AD9 00000000000606      <1>
347 00015AE0 000E0606060666663C- <1>     db  000h, 00eh, 006h, 006h, 006h, 006h, 066h, 066h, 03ch, 000h, 000h, 000h, 0e0h, 060h, 060h, 066h
347 00015AE9 000000E0606066      <1>
348 00015AF0 6C786C66E600000000- <1>     db  06ch, 078h, 06ch, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h, 038h, 018h, 018h, 018h, 018h, 018h
348 00015AF9 00381818181818      <1>
349 00015B00 18183C000000000000- <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ech, 0feh, 0d6h, 0d6h, 0d6h
349 00015B09 0000ECFED6D6D6      <1>
350 00015B10 C600000000000000000- <1>     db  0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 000h
350 00015B19 DC666666666600      <1>
351 00015B20 000000000000007CC6- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h
351 00015B29 C6C6C67C000000      <1>
352 00015B30 0000000000DC666666- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 07ch, 060h, 060h, 0f0h, 000h, 000h, 000h
352 00015B39 7C6060F0000000      <1>
353 00015B40 00000076CCCCCC7C0C- <1>     db  000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 07ch, 00ch, 00ch, 01eh, 000h, 000h, 000h, 000h, 000h
353 00015B49 0C1E0000000000      <1>
354 00015B50 00DC76666060F00000- <1>     db  000h, 0dch, 076h, 066h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch
354 00015B59 0000000000007C      <1>
355 00015B60 C6701CC67C00000000- <1>     db  0c6h, 070h, 01ch, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 010h, 030h, 030h, 0fch, 030h, 030h
355 00015B69 00103030FC3030      <1>
356 00015B70 30361C0000000000000- <1>     db  030h, 036h, 01ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch
356 00015B79 0000CCCCCCCCCC      <1>
357 00015B80 760000000000000000- <1>     db  076h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h, 03ch, 018h, 000h
357 00015B89 666666663C1800      <1>
358 00015B90 000000000000C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0d6h, 0d6h, 0feh, 06ch, 000h, 000h, 000h
358 00015B99 D6D6FE6C000000      <1>
359 00015BA0 0000000000C66C3838- <1>     db  000h, 000h, 000h, 000h, 000h, 0c6h, 06ch, 038h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h
```

```
359 00015BA9 6CC60000000000      <1>
360 00015BB0 000000C6C6C6C67E06- <1>     db  000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 0f8h, 000h, 000h, 000h, 000h, 000h
360 00015BB9 0CF80000000000      <1>
361 00015BC0 00FECC183066FE0000- <1>     db  000h, 0feh, 0cch, 018h, 030h, 066h, 0feh, 000h, 000h, 000h, 000h, 000h, 00eh, 018h, 018h, 018h
361 00015BC9 0000000E181818      <1>
362 00015BD0 701818180E00000000- <1>     db  070h, 018h, 018h, 018h, 00eh, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 000h, 018h
362 00015BD9 00181818180018      <1>
363 00015BE0 1818180000000000070- <1>     db  018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 070h, 018h, 018h, 018h, 00eh, 018h, 018h, 018h
363 00015BE9 1818180E181818      <1>
364 00015BF0 70000000000076DC00- <1>     db  070h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
364 00015BF9 00000000000000      <1>
365 00015C00 000000000000010386C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 000h, 000h, 000h, 000h
365 00015C09 C6C6FE00000000      <1>
366 00015C10 00003C66C2C0C0C266- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0c2h, 066h, 03ch, 00ch, 006h, 07ch, 000h, 000h, 000h
366 00015C19 3C0C067C000000      <1>
367 00015C20 CCCC00CCCCCCCCCC76- <1>     db  0cch, 0cch, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 00ch, 018h, 030h
367 00015C29 000000000C1830      <1>
368 00015C30 007CC6FEC0C67C0000- <1>     db  000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 000h, 078h
368 00015C39 000010386C0078      <1>
369 00015C40 0C7CCCCC7600000000- <1>     db  00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 0cch, 0cch, 000h, 078h, 00ch, 07ch
369 00015C49 00CCCC00780C7C      <1>
370 00015C50 CCCC76000000006030- <1>     db  0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 000h, 078h, 00ch, 07ch, 0cch, 0cch
370 00015C59 1800780C7CCCCC      <1>
371 00015C60 7600000000386C3800- <1>     db  076h, 000h, 000h, 000h, 000h, 038h, 06ch, 038h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 076h, 000h
371 00015C69 780C7CCCCC7600      <1>
372 00015C70 0000000000003C6660- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 03ch, 066h, 060h, 066h, 03ch, 00ch, 006h, 03ch, 000h, 000h
372 00015C79 663C0C063C0000      <1>
373 00015C80 0010386C007CC6FEC0- <1>     db  000h, 010h, 038h, 06ch, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h
373 00015C89 C67C0000000000      <1>
374 00015C90 CCCC007CC6FEC0C67C- <1>     db  0cch, 0cch, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 060h, 030h, 018h
374 00015C99 00000000603018      <1>
375 00015CA0 007CC6FEC0C67C0000- <1>     db  000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 000h, 038h
375 00015CA9 00000066660038      <1>
376 00015CB0 181818183C00000000- <1>     db  018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 018h, 03ch, 066h, 000h, 038h, 018h, 018h
376 00015CB9 183C6600381818      <1>
377 00015CC0 18183C000000006030- <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 000h, 038h, 018h, 018h, 018h, 018h
377 00015CC9 18003818181818      <1>
378 00015CD0 3C00000000C6C61038- <1>     db  03ch, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 000h
378 00015CD9 6CC6C6FEC6C600      <1>
379 00015CE0 0000386C3800386CC6- <1>     db  000h, 000h, 038h, 06ch, 038h, 000h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 000h, 000h, 000h
379 00015CE9 C6FEC6C6000000      <1>
380 00015CF0 18306000FE66607C60- <1>     db  018h, 030h, 060h, 000h, 0feh, 066h, 060h, 07ch, 060h, 066h, 0feh, 000h, 000h, 000h, 000h, 000h
380 00015CF9 66FE0000000000      <1>
381 00015D00 0000CC76367ED8D86E- <1>     db  000h, 000h, 0cch, 076h, 036h, 07eh, 0d8h, 0d8h, 06eh, 000h, 000h, 000h, 000h, 000h, 03eh, 06ch
381 00015D09 00000000003E6C      <1>
382 00015D10 CCCCFECCCCCCCE0000- <1>     db  0cch, 0cch, 0feh, 0cch, 0cch, 0cch, 0ceh, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 000h, 07ch
382 00015D19 000010386C007C      <1>
383 00015D20 C6C6C6C67C00000000- <1>     db  0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 000h, 07ch, 0c6h, 0c6h
383 00015D29 00C6C6007CC6C6      <1>
384 00015D30 C6C67C000000006030- <1>     db  0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h
384 00015D39 18007CC6C6C6C6      <1>
385 00015D40 7C000000003078CC00- <1>     db  07ch, 000h, 000h, 000h, 000h, 030h, 078h, 0cch, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h
385 00015D49 CCCCCCCCCC7600      <1>
386 00015D50 00000060301800CCCC- <1>     db  000h, 000h, 000h, 060h, 030h, 018h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h
386 00015D59 CCCCCC76000000      <1>
387 00015D60 0000C6C600C6C6C6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 078h, 000h, 000h, 0c6h
387 00015D69 7E060C780000C6      <1>
388 00015D70 C6386CC6C6C6C66C38- <1>     db  0c6h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 000h
388 00015D79 00000000C6C600      <1>
389 00015D80 C6C6C6C6C6C67C0000- <1>     db  0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 018h, 018h, 03ch, 066h, 060h
389 00015D89 000018183C6660      <1>
390 00015D90 60663C181800000000- <1>     db  060h, 066h, 03ch, 018h, 018h, 000h, 000h, 000h, 000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h
390 00015D99 386C6460F06060      <1>
391 00015DA0 60E6FC000000000066- <1>     db  060h, 0e6h, 0fch, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 03ch, 018h, 07eh, 018h, 07eh, 018h
391 00015DA9 663C187E187E18      <1>
392 00015DB0 1800000000F8CCCCF8- <1>     db  018h, 000h, 000h, 000h, 000h, 0f8h, 0cch, 0cch, 0f8h, 0c4h, 0cch, 0deh, 0cch, 0cch, 0c6h, 000h
392 00015DB9 C4CCDECCCCC600      <1>
393 00015DC0 0000000E1B1818187E- <1>     db  000h, 000h, 000h, 00eh, 01bh, 018h, 018h, 018h, 07eh, 018h, 018h, 018h, 018h, 0d8h, 070h, 000h
393 00015DC9 18181818D87000      <1>
394 00015DD0 0018306000780C7CCC- <1>     db  000h, 018h, 030h, 060h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 00ch
394 00015DD9 CC76000000000C      <1>
395 00015DE0 18300038181818183C- <1>     db  018h, 030h, 000h, 038h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 018h, 030h, 060h
395 00015DE9 00000000183060      <1>
396 00015DF0 007CC6C6C6C67C0000- <1>     db  000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 018h, 030h, 060h, 000h, 0cch
396 00015DF9 000018306000CC      <1>
397 00015E00 CCCCCCCC7600000000- <1>     db  0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 0dch, 066h, 066h
397 00015E09 0076DC00DC6666      <1>
398 00015E10 66666600000076DC00- <1>     db  066h, 066h, 066h, 000h, 000h, 000h, 076h, 0dch, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h
398 00015E19 C6E6F6FEDECEC6      <1>
399 00015E20 C6000000003C6C6C3E- <1>     db  0c6h, 000h, 000h, 000h, 000h, 03ch, 06ch, 06ch, 03eh, 000h, 07eh, 000h, 000h, 000h, 000h, 000h
399 00015E29 007E0000000000      <1>
400 00015E30 000000386C6C38007C- <1>     db  000h, 000h, 000h, 038h, 06ch, 06ch, 038h, 000h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h
400 00015E39 00000000000000      <1>
401 00015E40 0000303000303060C6- <1>     db  000h, 000h, 030h, 030h, 000h, 030h, 030h, 060h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h
401 00015E49 C67C0000000000      <1>
402 00015E50 00000000FEC0C0C000- <1>     db  000h, 000h, 000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
402 00015E59 00000000000000      <1>
403 00015E60 0000FE060606000000- <1>     db  000h, 000h, 0feh, 006h, 006h, 006h, 000h, 000h, 000h, 000h, 000h, 0c0h, 0c0h, 0c6h, 0cch, 0d8h
403 00015E69 0000C0C0C6CCD8      <1>
404 00015E70 3060DC860C183E0000- <1>     db  030h, 060h, 0dch, 086h, 00ch, 018h, 03eh, 000h, 000h, 0c0h, 0c0h, 0c6h, 0cch, 0d8h, 030h, 066h
404 00015E79 C0C0C6CCD83066      <1>
405 00015E80 CE9E3E060600000018- <1>     db  0ceh, 09eh, 03eh, 006h, 006h, 000h, 000h, 000h, 018h, 018h, 000h, 018h, 018h, 03ch, 03ch, 03ch
405 00015E89 180018183C3C3C      <1>
406 00015E90 180000000000000036- <1>     db  018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 036h, 06ch, 0d8h, 06ch, 036h, 000h, 000h, 000h
406 00015E99 6CD86C36000000      <1>
407 00015EA0 000000000000D86C36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0d8h, 06ch, 036h, 06ch, 0d8h, 000h, 000h, 000h, 000h, 000h
407 00015EA9 6CD800000000000     <1>
408 00015EB0 114414411441144411- <1>     db  011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 055h, 0aah
408 00015EB9 441144114455AA      <1>
409 00015EC0 55AA55AA55AA55AA55- <1>     db  055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 0ddh, 077h, 0ddh, 077h
409 00015EC9 AA55AADD77DD77      <1>
410 00015ED0 DD77DD77DD77DD77DD- <1>     db  0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 018h, 018h, 018h, 018h, 018h, 018h
410 00015ED9 77181818181818      <1>
411 00015EE0 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h
411 00015EE9 181818181818F8      <1>
412 00015EF0 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 018h, 018h
412 00015EF9 1818F818F81818      <1>
413 00015F00 1818181836363636- <1>     db  018h, 018h, 018h, 018h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0f6h, 036h, 036h, 036h, 036h
413 00015F09 3636F636363636      <1>
414 00015F10 3636000000000000- <1>     db  036h, 036h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 036h, 036h, 036h, 036h, 036h, 036h
414 00015F19 FE363636363636      <1>
415 00015F20 0000000000F818F818- <1>     db  000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 036h, 036h
415 00015F29 18181818183636      <1>
416 00015F30 363636F606F6363636- <1>     db  036h, 036h, 036h, 0f6h, 006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
416 00015F39 36363636363636      <1>
417 00015F40 3636363636363636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 000h, 000h, 000h, 000h, 000h, 0feh
417 00015F49 360000000000FE      <1>
418 00015F50 06F6363636363636- <1>     db  006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0f6h, 006h, 0feh
418 00015F59 36363636F606FE      <1>
419 00015F60 0000000000000363636- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0feh, 000h, 000h
419 00015F69 36363636FE0000      <1>
420 00015F70 0000000018181818- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 000h, 000h, 000h, 000h
420 00015F79 F818F800000000      <1>
421 00015F80 0000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h
```

```
421 00015F89 F8181818181818      <1>
422 00015F90 1818181818181F00-   <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h
422 00015F99 00000000001818      <1>
423 00015FA0 1818181818FF000000- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
423 00015FA9 00000000000000      <1>
424 00015FB0 000000FF1818181818- <1>     db  000h, 000h, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
424 00015FB9 18181818181818      <1>
425 00015FC0 181F18181818181800- <1>     db  018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh
425 00015FC9 000000000000FF      <1>
426 00015FD0 0000000000000181818- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 018h
426 00015FD9 18181818FF1818      <1>
427 00015FE0 1818181818181818181- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh, 018h, 018h, 018h
427 00015FE9 1F181F18181818      <1>
428 00015FF0 181836363636363636- <1>     db  018h, 018h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 037h, 036h, 036h, 036h, 036h, 036h, 036h
428 00015FF9 37363636363636      <1>
429 00016000 363636363637303F00- <1>     db  036h, 036h, 036h, 036h, 036h, 037h, 030h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
429 00016009 000000000000000     <1>
430 00016010 0000003F3037363636- <1>     db  000h, 000h, 000h, 03fh, 030h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
430 00016019 36363636363636      <1>
431 00016020 36F700FF0000000000- <1>     db  036h, 0f7h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh
431 00016029 000000000000FF      <1>
432 00016030 00F736363636363636- <1>     db  000h, 0f7h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 037h, 030h, 037h
432 00016039 36363636373037      <1>
433 00016040 363636363636000000- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 000h, 000h
433 00016049 0000FF00FF0000      <1>
434 00016050 0000000036363636363- <1>     db  000h, 000h, 000h, 000h, 036h, 036h, 036h, 036h, 036h, 0f7h, 000h, 0f7h, 036h, 036h, 036h, 036h
434 00016059 F700F736363636      <1>
435 00016060 363618181818FF00-   <1>     db  036h, 036h, 018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h
435 00016069 FF000000000000      <1>
436 00016070 36363636363636FF00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
436 00016079 000000000000000     <1>
437 00016080 000000FF00FF181818- <1>     db  000h, 000h, 000h, 0ffh, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h
437 00016089 18181800000000      <1>
438 00016090 000000FF3636363636- <1>     db  000h, 000h, 000h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
438 00016099 36363636363636      <1>
439 000160A0 363F000000000000018- <1>     db  036h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh
439 000160A9 181818181F181F      <1>
440 000160B0 0000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 01fh, 018h, 01fh, 018h, 018h
440 000160B9 00001F181F1818      <1>
441 000160C0 1818181800000000000- <1>     db  018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 03fh, 036h, 036h, 036h, 036h, 036h
441 000160C9 00003F36363636      <1>
442 000160D0 3636363636363636-   <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h
442 000160D9 FF363636363636      <1>
443 000160E0 1818181818FF18FF18- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
443 000160E9 18181818181818      <1>
444 000160F0 1818181818F8000000- <1>     db  018h, 018h, 018h, 018h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
444 000160F9 000000000000000     <1>
445 00016100 0000001F1818181818- <1>     db  000h, 000h, 000h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
445 00016109 18FFFFFFFFFFFF      <1>
446 00016110 FFFFFFFFFFFFFFFF00- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh
446 00016119 000000000000FF      <1>
447 00016120 FFFFFFFFFFFF0F0F0-  <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h
447 00016129 F0F0F0F0F0F0F0      <1>
448 00016130 F0F0F0F00F0F0F0F0F- <1>     db  0f0h, 0f0h, 0f0h, 0f0h, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh
448 00016139 0F0F0F0F0F0F0F      <1>
449 00016140 0F0FFFFFFFFFFFFF-   <1>     db  00fh, 00fh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h
449 00016149 000000000000000     <1>
450 00016150 000000000076DCD8D8- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0dch, 0d8h, 0d8h, 0dch, 076h, 000h, 000h, 000h, 000h, 000h
450 00016159 DC760000000000      <1>
451 00016160 00007CC6FCC6C6FCC0- <1>     db  000h, 000h, 07ch, 0c6h, 0fch, 0c6h, 0c6h, 0fch, 0c0h, 0c0h, 040h, 000h, 000h, 000h, 0feh, 0c6h
451 00016169 C040000000FEC6      <1>
452 00016170 C6C0C0C0C0C0C00000- <1>     db  0c6h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 06ch
452 00016179 0000000000FE6C      <1>
453 00016180 6C6C6C6C6C00000000- <1>     db  06ch, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 000h, 000h, 0feh, 0c6h, 060h, 030h, 018h, 030h
453 00016189 00FEC660301830      <1>
454 00016190 60C6FE000000000000- <1>     db  060h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 0d8h, 0d8h, 0d8h, 0d8h
454 00016199 00007ED8D8D8D8      <1>
455 000161A0 700000000000000066- <1>     db  070h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h, 07ch, 060h, 060h, 0c0h
455 000161A9 6666667C6060C0      <1>
456 000161B0 00000000000076DC18- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h
456 000161B9 18181818000000      <1>
457 000161C0 00007E183C6666663C- <1>     db  000h, 000h, 07eh, 018h, 03ch, 066h, 066h, 066h, 03ch, 018h, 07eh, 000h, 000h, 000h, 000h, 000h
457 000161C9 187E0000000000      <1>
458 000161D0 386CC6C6FEC6C66C38- <1>     db  038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 038h, 06ch
458 000161D9 0000000000386C      <1>
459 000161E0 C6C6C66C6C6CEE0000- <1>     db  0c6h, 0c6h, 0c6h, 06ch, 06ch, 06ch, 0eeh, 000h, 000h, 000h, 000h, 000h, 01eh, 030h, 018h, 00ch
459 000161E9 0000001E30180C      <1>
460 000161F0 3E6666663C00000000- <1>     db  03eh, 066h, 066h, 066h, 03ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 0dbh, 0dbh
460 000161F9 000000007EDBDB      <1>
461 00016200 7E0000000000000003- <1>     db  07eh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 003h, 006h, 07eh, 0dbh, 0dbh, 0f3h, 07eh, 060h
461 00016209 067EDBDBF37E60      <1>
462 00016210 C0000000001C3060-   <1>     db  0c0h, 000h, 000h, 000h, 000h, 000h, 01ch, 030h, 060h, 060h, 07ch, 060h, 060h, 030h, 01ch, 000h
462 00016219 607C6060301C00      <1>
463 00016220 00000000007CC6C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h
463 00016229 C6C6C6C6000000      <1>
464 00016230 000000FE0000FE0000- <1>     db  000h, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h
464 00016239 FE000000000000      <1>
465 00016240 0018187E18180000FF- <1>     db  000h, 018h, 018h, 07eh, 018h, 018h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 030h, 018h
465 00016249 00000000003018      <1>
466 00016250 0C060C1830007E0000- <1>     db  00ch, 006h, 00ch, 018h, 030h, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 00ch, 018h, 030h, 060h
466 00016259 0000000C183060      <1>
467 00016260 30180C007E00000000- <1>     db  030h, 018h, 00ch, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 00eh, 01bh, 01bh, 018h, 018h, 018h
467 00016269 000E1B1B181818      <1>
468 00016270 1818181818181818-   <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0d8h, 0d8h
468 00016279 1818181818D8D8      <1>
469 00016280 7000000000000001818- <1>     db  070h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 07eh, 000h, 018h, 018h, 000h, 000h
469 00016289 007E0018180000      <1>
470 00016290 00000000000076DC00- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h
470 00016299 76DC0000000000      <1>
471 000162A0 00386C6C3800000000- <1>     db  000h, 038h, 06ch, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
471 000162A9 000000000000000     <1>
472 000162B0 0000000018180000000- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
472 000162B9 000000000000000     <1>
473 000162C0 0000001800000000000- <1>     db  000h, 000h, 000h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 00fh, 00ch, 00ch, 00ch, 00ch
473 000162C9 00000F0C0C0C0C      <1>
474 000162D0 0CEC6C3C1C00000000- <1>     db  00ch, 0ech, 06ch, 03ch, 01ch, 000h, 000h, 000h, 000h, 0d8h, 06ch, 06ch, 06ch, 06ch, 06ch, 000h
474 000162D9 D86C6C6C6C6C00      <1>
475 000162E0 0000000000000070D8- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 070h, 0d8h, 030h, 060h, 0c8h, 0f8h, 000h, 000h, 000h
475 000162E9 3060C8F8000000      <1>
476 000162F0 00000000000000007C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch, 07ch, 07ch, 07ch, 07ch, 07ch, 000h, 000h
476 000162F9 7C7C7C7C7C0000      <1>
477 00016300 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
477 00016309 000000000000000     <1>
478                              <1> vgafont16:
479 00016310 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
479 00016319 000000000000000     <1>
480 00016320 00007E81A58181BD99- <1>     db  000h, 000h, 07eh, 081h, 0a5h, 081h, 081h, 0bdh, 099h, 081h, 081h, 07eh, 000h, 000h, 000h, 000h
480 00016329 81817E00000000      <1>
481 00016330 00007EFFDBFFFFC3E7- <1>     db  000h, 000h, 07eh, 0ffh, 0dbh, 0ffh, 0ffh, 0c3h, 0e7h, 0ffh, 0ffh, 07eh, 000h, 000h, 000h, 000h
481 00016339 FFFF7E00000000      <1>
482 00016340 000000006CFEFEFEFE- <1>     db  000h, 000h, 000h, 000h, 06ch, 0feh, 0feh, 0feh, 0feh, 07ch, 038h, 010h, 000h, 000h, 000h, 000h
482 00016349 7C3810000000000     <1>
483 00016350 0000000010387CFE7C- <1>     db  000h, 000h, 000h, 000h, 010h, 038h, 07ch, 0feh, 07ch, 038h, 010h, 000h, 000h, 000h, 000h, 000h
483 00016359 38100000000000      <1>
```

```
484 00016360 000000183C3CE7E7E7- <1>     db  000h, 000h, 000h, 018h, 03ch, 03ch, 0e7h, 0e7h, 0e7h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
484 00016369 18183C00000000      <1>
485 00016370 000000183C7EFFFF7E- <1>     db  000h, 000h, 000h, 018h, 03ch, 07eh, 0ffh, 0ffh, 07eh, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
485 00016379 18183C00000000      <1>
486 00016380 000000000000183C3C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 03ch, 018h, 000h, 000h, 000h, 000h, 000h, 000h
486 00016389 18000000000000      <1>
487 00016390 FFFFFFFFFFFFE7C3C3- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0e7h, 0c3h, 0c3h, 0e7h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
487 00016399 E7FFFFFFFFFFFF      <1>
488 000163A0 00000000003C664242- <1>     db  000h, 000h, 000h, 000h, 000h, 03ch, 066h, 042h, 042h, 066h, 03ch, 000h, 000h, 000h, 000h, 000h
488 000163A9 663C0000000000      <1>
489 000163B0 FFFFFFFFFFC399BDBD- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0c3h, 099h, 0bdh, 0bdh, 099h, 0c3h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
489 000163B9 99C3FFFFFFFFFF      <1>
490 000163C0 00001E0E1A3278CCCC- <1>     db  000h, 000h, 01eh, 00eh, 01ah, 032h, 078h, 0cch, 0cch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h
490 000163C9 CCCC7800000000      <1>
491 000163D0 00003C666666663C18- <1>     db  000h, 000h, 03ch, 066h, 066h, 066h, 066h, 03ch, 018h, 07eh, 018h, 018h, 000h, 000h, 000h, 000h
491 000163D9 7E181800000000      <1>
492 000163E0 00003F333F30303030- <1>     db  000h, 000h, 03fh, 033h, 03fh, 030h, 030h, 030h, 030h, 070h, 0f0h, 0e0h, 000h, 000h, 000h, 000h
492 000163E9 70F0E000000000      <1>
493 000163F0 00007F637F63636363- <1>     db  000h, 000h, 07fh, 063h, 07fh, 063h, 063h, 063h, 063h, 067h, 0e7h, 0e6h, 0c0h, 000h, 000h, 000h
493 000163F9 67E7E6C0000000      <1>
494 00016400 0000001818DB3CE73C- <1>     db  000h, 000h, 000h, 018h, 018h, 0dbh, 03ch, 0e7h, 03ch, 0dbh, 018h, 018h, 000h, 000h, 000h, 000h
494 00016409 DB181800000000      <1>
495 00016410 0080C0E0F0F8FEF8F0- <1>     db  000h, 080h, 0c0h, 0e0h, 0f0h, 0f8h, 0feh, 0f8h, 0f0h, 0e0h, 0c0h, 080h, 000h, 000h, 000h, 000h
495 00016419 E0C080000000000     <1>
496 00016420 0002060E1E3EFE3E1E- <1>     db  000h, 002h, 006h, 00eh, 01eh, 03eh, 0feh, 03eh, 01eh, 00eh, 006h, 002h, 000h, 000h, 000h, 000h
496 00016429 0E060200000000      <1>
497 00016430 0000183C7E1818187E- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h, 000h
497 00016439 3C180000000000      <1>
498 00016440 0000666666666666- <1>     db  000h, 000h, 066h, 066h, 066h, 066h, 066h, 066h, 066h, 000h, 066h, 066h, 000h, 000h, 000h, 000h
498 00016449 00666600000000      <1>
499 00016450 00007FDBDBDB7B1B1B- <1>     db  000h, 000h, 07fh, 0dbh, 0dbh, 0dbh, 07bh, 01bh, 01bh, 01bh, 01bh, 01bh, 000h, 000h, 000h, 000h
499 00016459 1B1B1B00000000      <1>
500 00016460 007CC660386CC6C66C- <1>     db  000h, 07ch, 0c6h, 060h, 038h, 06ch, 0c6h, 0c6h, 06ch, 038h, 00ch, 0c6h, 07ch, 000h, 000h, 000h
500 00016469 380CC67C000000      <1>
501 00016470 00000000000000FE- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 0feh, 0feh, 0feh, 000h, 000h, 000h, 000h
501 00016479 FEFEFE00000000      <1>
502 00016480 0000183C7E1818187E- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 07eh, 000h, 000h, 000h, 000h
502 00016489 3C187E00000000      <1>
503 00016490 0000183C7E18181818- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h
503 00016499 18181800000000      <1>
504 000164A0 0000181818181818- <1>     db  000h, 000h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h
504 000164A9 7E3C1800000000      <1>
505 000164B0 0000000000180CFE0C- <1>     db  000h, 000h, 000h, 000h, 000h, 018h, 00ch, 0feh, 00ch, 018h, 000h, 000h, 000h, 000h, 000h, 000h
505 000164B9 18000000000000      <1>
506 000164C0 00000000003060FE60- <1>     db  000h, 000h, 000h, 000h, 000h, 030h, 060h, 0feh, 060h, 030h, 000h, 000h, 000h, 000h, 000h, 000h
506 000164C9 30000000000000      <1>
507 000164D0 000000000000C0C0C0- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0c0h, 0c0h, 0c0h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h
507 000164D9 FE000000000000      <1>
508 000164E0 00000000002466FF66- <1>     db  000h, 000h, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h, 000h, 000h, 000h, 000h
508 000164E9 24000000000000      <1>
509 000164F0 0000000001038387C7C- <1>     db  000h, 000h, 000h, 000h, 010h, 038h, 038h, 07ch, 07ch, 0feh, 0feh, 000h, 000h, 000h, 000h, 000h
509 000164F9 FEFE0000000000      <1>
510 00016500 00000000FEFE7C7C38- <1>     db  000h, 000h, 000h, 000h, 0feh, 0feh, 07ch, 07ch, 038h, 038h, 010h, 000h, 000h, 000h, 000h, 000h
510 00016509 38100000000000      <1>
511 00016510 0000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
511 00016519 00000000000000      <1>
512 00016520 0000183C3C181818- <1>     db  000h, 000h, 018h, 03ch, 03ch, 03ch, 018h, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h, 000h
512 00016529 00181800000000      <1>
513 00016530 006666662400000000- <1>     db  000h, 066h, 066h, 066h, 024h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
513 00016539 00000000000000      <1>
514 00016540 0000006C6CFE6C6C6C- <1>     db  000h, 000h, 000h, 06ch, 06ch, 0feh, 06ch, 06ch, 06ch, 0feh, 06ch, 06ch, 000h, 000h, 000h, 000h
514 00016549 FE6C6C00000000      <1>
515 00016550 18187CC6C2C07C0606- <1>     db  018h, 018h, 07ch, 0c6h, 0c2h, 0c0h, 07ch, 006h, 006h, 086h, 0c6h, 07ch, 018h, 018h, 000h, 000h
515 00016559 86C67C18180000      <1>
516 00016560 00000000C2C60C1830- <1>     db  000h, 000h, 000h, 000h, 0c2h, 0c6h, 00ch, 018h, 030h, 060h, 0c6h, 086h, 000h, 000h, 000h, 000h
516 00016569 60C68600000000      <1>
517 00016570 0000386C6C3876DCCC- <1>     db  000h, 000h, 038h, 06ch, 06ch, 038h, 076h, 0dch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
517 00016579 CCCC7600000000      <1>
518 00016580 003030306000000000- <1>     db  000h, 030h, 030h, 030h, 060h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
518 00016589 00000000000000      <1>
519 00016590 00000C183030303030- <1>     db  000h, 000h, 00ch, 018h, 030h, 030h, 030h, 030h, 030h, 030h, 018h, 00ch, 000h, 000h, 000h, 000h
519 00016599 30180C00000000      <1>
520 000165A0 000030180C0C0C0C0C- <1>     db  000h, 000h, 030h, 018h, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 018h, 030h, 000h, 000h, 000h, 000h
520 000165A9 0C183000000000      <1>
521 000165B0 0000000000663CFF3C- <1>     db  000h, 000h, 000h, 000h, 000h, 066h, 03ch, 0ffh, 03ch, 066h, 000h, 000h, 000h, 000h, 000h, 000h
521 000165B9 66000000000000      <1>
522 000165C0 000000000018187E18- <1>     db  000h, 000h, 000h, 000h, 000h, 018h, 018h, 07eh, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h
522 000165C9 18000000000000      <1>
523 000165D0 0000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 030h, 000h, 000h, 000h
523 000165D9 18181830000000      <1>
524 000165E0 00000000000000FE00- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
524 000165E9 00000000000000      <1>
525 000165F0 0000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h
525 000165F9 00181800000000      <1>
526 00016600 0000000002060C1830- <1>     db  000h, 000h, 000h, 000h, 002h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 080h, 000h, 000h, 000h, 000h
526 00016609 60C08000000000      <1>
527 00016610 00003C66C3C3DBDBC3- <1>     db  000h, 000h, 03ch, 066h, 0c3h, 0c3h, 0dbh, 0dbh, 0c3h, 0c3h, 066h, 03ch, 000h, 000h, 000h, 000h
527 00016619 C3663C00000000      <1>
528 00016620 0000183878181818- <1>     db  000h, 000h, 018h, 038h, 078h, 018h, 018h, 018h, 018h, 018h, 018h, 07eh, 000h, 000h, 000h, 000h
528 00016629 18187E00000000      <1>
529 00016630 00007CC6060C183060- <1>     db  000h, 000h, 07ch, 0c6h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 0c6h, 0feh, 000h, 000h, 000h, 000h
529 00016639 C0C6FE00000000      <1>
530 00016640 00007CC606063C0606- <1>     db  000h, 000h, 07ch, 0c6h, 006h, 006h, 03ch, 006h, 006h, 006h, 0c6h, 07ch, 000h, 000h, 000h, 000h
530 00016649 06C67C00000000      <1>
531 00016650 00000C1C3C6CCCFE0C- <1>     db  000h, 000h, 00ch, 01ch, 03ch, 06ch, 0cch, 0feh, 00ch, 00ch, 00ch, 01eh, 000h, 000h, 000h, 000h
531 00016659 0C0C1E00000000      <1>
532 00016660 0000FEC0C0C0FC0606- <1>     db  000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 0fch, 006h, 006h, 006h, 0c6h, 07ch, 000h, 000h, 000h, 000h
532 00016669 06C67C00000000      <1>
533 00016670 00003860C0C0FCC6C6- <1>     db  000h, 000h, 038h, 060h, 0c0h, 0c0h, 0fch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
533 00016679 C6C67C00000000      <1>
534 00016680 0000FEC606060C1830- <1>     db  000h, 000h, 0feh, 0c6h, 006h, 006h, 00ch, 018h, 030h, 030h, 030h, 030h, 000h, 000h, 000h, 000h
534 00016689 30303000000000      <1>
535 00016690 00007CC6C6C67CC6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
535 00016699 C6C67C00000000      <1>
536 000166A0 00007CC6C6C67E0606- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 07eh, 006h, 006h, 006h, 00ch, 078h, 000h, 000h, 000h, 000h
536 000166A9 060C7800000000      <1>
537 000166B0 0000000018180000000- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h
537 000166B9 18180000000000      <1>
538 000166C0 0000000018180000000- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 018h, 018h, 030h, 000h, 000h, 000h, 000h
538 000166C9 18183000000000      <1>
539 000166D0 000000060C18306030- <1>     db  000h, 000h, 000h, 006h, 00ch, 018h, 030h, 060h, 030h, 018h, 00ch, 006h, 000h, 000h, 000h, 000h
539 000166D9 180C0600000000      <1>
540 000166E0 00000000007E00007E- <1>     db  000h, 000h, 000h, 000h, 000h, 07eh, 000h, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 000h, 000h
540 000166E9 00000000000000      <1>
541 000166F0 0000006030180C060C- <1>     db  000h, 000h, 000h, 060h, 030h, 018h, 00ch, 006h, 00ch, 018h, 030h, 060h, 000h, 000h, 000h, 000h
541 000166F9 18306000000000      <1>
542 00016700 00007CC6C60C181818- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 00ch, 018h, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h, 000h
542 00016709 00181800000000      <1>
543 00016710 0000007CC6C6DEDEDE- <1>     db  000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0deh, 0deh, 0deh, 0dch, 0c0h, 07ch, 000h, 000h, 000h, 000h
543 00016719 DCC07C00000000      <1>
544 00016720 000010386CC6C6FEC6- <1>     db  000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
544 00016729 C6C6C600000000      <1>
545 00016730 0000FC6666667C6666- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 066h, 066h, 066h, 066h, 0fch, 000h, 000h, 000h, 000h
545 00016739 6666FC00000000      <1>
```

```
546 00016740 00003C66C2C0C0C0C0- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0c0h, 0c0h, 0c2h, 066h, 03ch, 000h, 000h, 000h, 000h
546 00016749 C2663C00000000      <1>
547 00016750 0000F86C6666666666- <1>     db  000h, 000h, 0f8h, 06ch, 066h, 066h, 066h, 066h, 066h, 066h, 06ch, 0f8h, 000h, 000h, 000h, 000h
547 00016759 666CF800000000      <1>
548 00016760 0000FE666268786860- <1>     db  000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 060h, 062h, 066h, 0feh, 000h, 000h, 000h, 000h
548 00016769 6266FE00000000      <1>
549 00016770 0000FE666268786860- <1>     db  000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h
549 00016779 6060F000000000      <1>
550 00016780 00003C66C2C0C0DEC6- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0deh, 0c6h, 0c6h, 066h, 03ah, 000h, 000h, 000h, 000h
550 00016789 C6663A00000000      <1>
551 00016790 0000C6C6C6C6FEC6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
551 00016799 C6C6C600000000      <1>
552 000167A0 00003C181818181818- <1>     db  000h, 000h, 03ch, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
552 000167A9 18183C00000000      <1>
553 000167B0 00001E0C0C0C0C0CCC- <1>     db  000h, 000h, 01eh, 00ch, 00ch, 00ch, 00ch, 00ch, 0cch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h
553 000167B9 CCCC7800000000      <1>
554 000167C0 0000E666666C78786C- <1>     db  000h, 000h, 0e6h, 066h, 066h, 06ch, 078h, 078h, 06ch, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h
554 000167C9 6666E600000000      <1>
555 000167D0 0000F0606060606060- <1>     db  000h, 000h, 0f0h, 060h, 060h, 060h, 060h, 060h, 060h, 062h, 066h, 0feh, 000h, 000h, 000h, 000h
555 000167D9 6266FE00000000      <1>
556 000167E0 0000C3E7FFFFDBC3C3- <1>     db  000h, 000h, 0c3h, 0e7h, 0ffh, 0ffh, 0dbh, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 000h, 000h, 000h, 000h
556 000167E9 C3C3C300000000      <1>
557 000167F0 0000C6E6F6FEDECEC6- <1>     db  000h, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
557 000167F9 C6C6C600000000      <1>
558 00016800 00007CC6C6C6C6C6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
558 00016809 C6C67C00000000      <1>
559 00016810 0000FC6666667C6060- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 060h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h
559 00016819 6060F000000000      <1>
560 00016820 00007CC6C6C6C6C6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0d6h, 0deh, 07ch, 00ch, 00eh, 000h, 000h
560 00016829 D6DE7C0C0E0000      <1>
561 00016830 0000FC6666667C6C66- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 06ch, 066h, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h
561 00016839 6666E600000000      <1>
562 00016840 00007CC6C660380C06- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 060h, 038h, 00ch, 006h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
562 00016849 C6C67C00000000      <1>
563 00016850 0000FFDB9918181818- <1>     db  000h, 000h, 0ffh, 0dbh, 099h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
563 00016859 18183C00000000      <1>
564 00016860 0000C6C6C6C6C6C6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
564 00016869 C6C67C00000000      <1>
565 00016870 0000C3C3C3C3C3C3C3- <1>     db  000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h, 000h
565 00016879 663C1800000000      <1>
566 00016880 0000C3C3C3C3C3DBDB- <1>     db  000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 066h, 000h, 000h, 000h, 000h
566 00016889 FF666600000000      <1>
567 00016890 0000C3C3663C18183C- <1>     db  000h, 000h, 0c3h, 0c3h, 066h, 03ch, 018h, 018h, 03ch, 066h, 0c3h, 0c3h, 000h, 000h, 000h, 000h
567 00016899 66C3C300000000      <1>
568 000168A0 0000C3C3C3663C1818- <1>     db  000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
568 000168A9 18183C00000000      <1>
569 000168B0 0000FFC3860C183060- <1>     db  000h, 000h, 0ffh, 0c3h, 086h, 00ch, 018h, 030h, 060h, 0c1h, 0c3h, 0ffh, 000h, 000h, 000h, 000h
569 000168B9 C1C3FF00000000      <1>
570 000168C0 00003C303030303030- <1>     db  000h, 000h, 03ch, 030h, 030h, 030h, 030h, 030h, 030h, 030h, 030h, 03ch, 000h, 000h, 000h, 000h
570 000168C9 30303C00000000      <1>
571 000168D0 00000080C0E070381C- <1>     db  000h, 000h, 000h, 080h, 0c0h, 0e0h, 070h, 038h, 01ch, 00eh, 006h, 002h, 000h, 000h, 000h, 000h
571 000168D9 0E0602000000000     <1>
572 000168E0 00003C0C0C0C0C0C0C- <1>     db  000h, 000h, 03ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 03ch, 000h, 000h, 000h, 000h
572 000168E9 0C0C3C00000000      <1>
573 000168F0 10386CC600000000000- <1>     db  010h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
573 000168F9 00000000000000      <1>
574 00016900 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 000h
574 00016909 00000000FF0000      <1>
575 00016910 3030180000000000000- <1>     db  030h, 030h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
575 00016919 00000000000000      <1>
576 00016920 0000000000780C7CCC- <1>     db  000h, 000h, 000h, 000h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
576 00016929 CCCC7600000000      <1>
577 00016930 0000E06060786C6666- <1>     db  000h, 000h, 0e0h, 060h, 060h, 078h, 06ch, 066h, 066h, 066h, 066h, 07ch, 000h, 000h, 000h, 000h
577 00016939 66667C00000000      <1>
578 00016940 00000000007CC6C0C0- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c0h, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
578 00016949 C0C67C00000000      <1>
579 00016950 00001C0C0C3C6CCCCC- <1>     db  000h, 000h, 01ch, 00ch, 00ch, 03ch, 06ch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
579 00016959 CCCC7600000000      <1>
580 00016960 00000000007CC6FEC0- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
580 00016969 C0C67C00000000      <1>
581 00016970 0000386C6460F06060- <1>     db  000h, 000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h
581 00016979 6060F000000000      <1>
582 00016980 000000000076CCCCCC- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 0cch, 0cch, 07ch, 00ch, 0cch, 078h, 000h
582 00016989 CCCC7C0CCC7800      <1>
583 00016990 0000E060606C766666- <1>     db  000h, 000h, 0e0h, 060h, 060h, 06ch, 076h, 066h, 066h, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h
583 00016999 6666E600000000      <1>
584 000169A0 000018180038181818- <1>     db  000h, 000h, 018h, 018h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
584 000169A9 18183C00000000      <1>
585 000169B0 00000606000E060606- <1>     db  000h, 000h, 006h, 006h, 000h, 00eh, 006h, 006h, 006h, 006h, 006h, 006h, 066h, 066h, 03ch, 000h
585 000169B9 06060666663C00      <1>
586 000169C0 0000E06060666C7878- <1>     db  000h, 000h, 0e0h, 060h, 060h, 066h, 06ch, 078h, 078h, 06ch, 066h, 0e6h, 000h, 000h, 000h, 000h
586 000169C9 6C66E600000000      <1>
587 000169D0 000038181818181818- <1>     db  000h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
587 000169D9 18183C00000000      <1>
588 000169E0 0000000000E6FFDBDB- <1>     db  000h, 000h, 000h, 000h, 000h, 0e6h, 0ffh, 0dbh, 0dbh, 0dbh, 0dbh, 0dbh, 000h, 000h, 000h, 000h
588 000169E9 DBDBDB00000000      <1>
589 000169F0 0000000000DC666666- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 066h, 000h, 000h, 000h, 000h
589 000169F9 66666600000000      <1>
590 00016A00 00000000007CC6C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
590 00016A09 C6C67C00000000      <1>
591 00016A10 0000000000DC666666- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 07ch, 060h, 060h, 0f0h, 000h
591 00016A19 66667C6060F000      <1>
592 00016A20 000000000076CCCCCC- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 0cch, 0cch, 07ch, 00ch, 00ch, 01eh, 000h
592 00016A29 CCCC7C0C0C1E00      <1>
593 00016A30 0000000000DC766660- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 076h, 066h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h
593 00016A39 6060F000000000      <1>
594 00016A40 00000000007CC66038- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 060h, 038h, 00ch, 0c6h, 07ch, 000h, 000h, 000h, 000h
594 00016A49 0CC67C00000000      <1>
595 00016A50 0000103030FC303030- <1>     db  000h, 000h, 010h, 030h, 030h, 0fch, 030h, 030h, 030h, 030h, 036h, 01ch, 000h, 000h, 000h, 000h
595 00016A59 30361C00000000      <1>
596 00016A60 0000000000CCCCCCCC- <1>     db  000h, 000h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
596 00016A69 CCCC7600000000      <1>
597 00016A70 0000000000C3C3C3C3- <1>     db  000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h, 000h
597 00016A79 663C1800000000      <1>
598 00016A80 0000000000C3C3C3DB- <1>     db  000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 000h, 000h, 000h, 000h
598 00016A89 DBFF6600000000      <1>
599 00016A90 0000000000C3663C18- <1>     db  000h, 000h, 000h, 000h, 000h, 0c3h, 066h, 03ch, 018h, 03ch, 066h, 0c3h, 000h, 000h, 000h, 000h
599 00016A99 3C66C300000000      <1>
600 00016AA0 0000000000C6C6C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 0f8h, 000h
600 00016AA9 C6C67E060CF800      <1>
601 00016AB0 0000000000FECC1830- <1>     db  000h, 000h, 000h, 000h, 000h, 0feh, 0cch, 018h, 030h, 060h, 0c6h, 0feh, 000h, 000h, 000h, 000h
601 00016AB9 60C6FE00000000      <1>
602 00016AC0 00000E181818701818- <1>     db  000h, 000h, 00eh, 018h, 018h, 018h, 070h, 018h, 018h, 018h, 018h, 00eh, 000h, 000h, 000h, 000h
602 00016AC9 18180E00000000      <1>
603 00016AD0 0000181818180018 18- <1>     db  000h, 000h, 018h, 018h, 018h, 018h, 000h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h
603 00016AD9 18181800000000      <1>
604 00016AE0 0000701818180E1818- <1>     db  000h, 000h, 070h, 018h, 018h, 018h, 00eh, 018h, 018h, 018h, 018h, 070h, 000h, 000h, 000h, 000h
604 00016AE9 18187000000000      <1>
605 00016AF0 000076DC0000000000- <1>     db  000h, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
605 00016AF9 00000000000000      <1>
606 00016B00 0000000010386CC6C6- <1>     db  000h, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h
606 00016B09 C6FE0000000000      <1>
607 00016B10 00003C66C2C0C0C0C2- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0c0h, 0c2h, 066h, 03ch, 00ch, 006h, 07ch, 000h, 000h
607 00016B19 663C0C067C0000      <1>
```

```
608 00016B20 0000CC0000CCCCCCCC- <1>     db  000h, 000h, 0cch, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
608 00016B29 CCCC7600000000      <1>
609 00016B30 000C1830007CC6FEC0- <1>     db  000h, 00ch, 018h, 030h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
609 00016B39 C0C67C00000000      <1>
610 00016B40 0010386C00780C7CCC- <1>     db  000h, 010h, 038h, 06ch, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
610 00016B49 CCCC7600000000      <1>
611 00016B50 0000CC0000780C7CCC- <1>     db  000h, 000h, 0cch, 000h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
611 00016B59 CCCC7600000000      <1>
612 00016B60 0060301800780C7CCC- <1>     db  000h, 060h, 030h, 018h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
612 00016B69 CCCC7600000000      <1>
613 00016B70 00386C3800780C7CCC- <1>     db  000h, 038h, 06ch, 038h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
613 00016B79 CCCC7600000000      <1>
614 00016B80 000000003C66606066- <1>     db  000h, 000h, 000h, 000h, 03ch, 066h, 060h, 060h, 066h, 03ch, 00ch, 006h, 03ch, 000h, 000h, 000h
614 00016B89 3C0C063C000000      <1>
615 00016B90 0010386C007CC6FEC0- <1>     db  000h, 010h, 038h, 06ch, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
615 00016B99 C0C67C00000000      <1>
616 00016BA0 0000C600007CC6FEC0- <1>     db  000h, 000h, 0c6h, 000h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
616 00016BA9 C0C67C00000000      <1>
617 00016BB0 00603018007CC6FEC0- <1>     db  000h, 060h, 030h, 018h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
617 00016BB9 C0C67C00000000      <1>
618 00016BC0 000066000038181818- <1>     db  000h, 000h, 066h, 000h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
618 00016BC9 18183C00000000      <1>
619 00016BD0 00183C660038181818- <1>     db  000h, 018h, 03ch, 066h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
619 00016BD9 18183C00000000      <1>
620 00016BE0 006030180038181818- <1>     db  000h, 060h, 030h, 018h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
620 00016BE9 18183C00000000      <1>
621 00016BF0 00C60010386CC6C6FE- <1>     db  000h, 0c6h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
621 00016BF9 C6C6C600000000      <1>
622 00016C00 386C3800386CC6C6FE- <1>     db  038h, 06ch, 038h, 000h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
622 00016C09 C6C6C600000000      <1>
623 00016C10 18306000FE66607C60- <1>     db  018h, 030h, 060h, 000h, 0feh, 066h, 060h, 07ch, 060h, 060h, 066h, 0feh, 000h, 000h, 000h, 000h
623 00016C19 6066FE00000000      <1>
624 00016C20 00000000006E3B1B7E- <1>     db  000h, 000h, 000h, 000h, 000h, 06eh, 03bh, 01bh, 07eh, 0d8h, 0dch, 077h, 000h, 000h, 000h, 000h
624 00016C29 D8DC7700000000      <1>
625 00016C30 00003E6CCCCCFECCCC- <1>     db  000h, 000h, 03eh, 06ch, 0cch, 0cch, 0feh, 0cch, 0cch, 0cch, 0cch, 0ceh, 000h, 000h, 000h, 000h
625 00016C39 CCCCCE00000000      <1>
626 00016C40 0010386C007CC6C6C6- <1>     db  000h, 010h, 038h, 06ch, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
626 00016C49 C6C67C00000000      <1>
627 00016C50 0000C600007CC6C6C6- <1>     db  000h, 000h, 0c6h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
627 00016C59 C6C67C00000000      <1>
628 00016C60 00603018007CC6C6C6- <1>     db  000h, 060h, 030h, 018h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
628 00016C69 C6C67C00000000      <1>
629 00016C70 003078CC00CCCCCCCC- <1>     db  000h, 030h, 078h, 0cch, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
629 00016C79 CCCC7600000000      <1>
630 00016C80 0060301800CCCCCCCC- <1>     db  000h, 060h, 030h, 018h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
630 00016C89 CCCC7600000000      <1>
631 00016C90 0000C60000C6C6C6C6- <1>     db  000h, 000h, 0c6h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 078h, 000h
631 00016C99 C6C67E060C7800      <1>
632 00016CA0 00C6007CC6C6C6C6C6- <1>     db  000h, 0c6h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
632 00016CA9 C6C67C00000000      <1>
633 00016CB0 00C600C6C6C6C6C6C6- <1>     db  000h, 0c6h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
633 00016CB9 C6C67C00000000      <1>
634 00016CC0 0018187EC3C0C0C0C3- <1>     db  000h, 018h, 018h, 07eh, 0c3h, 0c0h, 0c0h, 0c0h, 0c3h, 07eh, 018h, 018h, 000h, 000h, 000h, 000h
634 00016CC9 7E181800000000      <1>
635 00016CD0 00386C6460F0606060- <1>     db  000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h, 060h, 060h, 0e6h, 0fch, 000h, 000h, 000h, 000h
635 00016CD9 60E6FC00000000      <1>
636 00016CE0 0000C3663C18FF18FF- <1>     db  000h, 000h, 0c3h, 066h, 03ch, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 000h, 000h, 000h, 000h
636 00016CE9 18181800000000      <1>
637 00016CF0 00FC66667C62666F66- <1>     db  000h, 0fch, 066h, 066h, 07ch, 062h, 066h, 06fh, 066h, 066h, 066h, 0f3h, 000h, 000h, 000h, 000h
637 00016CF9 6666F300000000      <1>
638 00016D00 000E1B1818187E1818- <1>     db  000h, 00eh, 01bh, 018h, 018h, 018h, 07eh, 018h, 018h, 018h, 018h, 018h, 0d8h, 070h, 000h, 000h
638 00016D09 181818D8700000      <1>
639 00016D10 0018306000780C7CCC- <1>     db  000h, 018h, 030h, 060h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
639 00016D19 CCCC7600000000      <1>
640 00016D20 000C18300038181818- <1>     db  000h, 00ch, 018h, 030h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
640 00016D29 18183C00000000      <1>
641 00016D30 00183060007CC6C6C6- <1>     db  000h, 018h, 030h, 060h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
641 00016D39 C6C67C00000000      <1>
642 00016D40 0018306000CCCCCCCC- <1>     db  000h, 018h, 030h, 060h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
642 00016D49 CCCC7600000000      <1>
643 00016D50 000076DC00DC666666- <1>     db  000h, 000h, 076h, 0dch, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 066h, 000h, 000h, 000h, 000h
643 00016D59 66666600000000      <1>
644 00016D60 76DC00C6E6F6FEDECE- <1>     db  076h, 0dch, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
644 00016D69 C6C6C600000000      <1>
645 00016D70 003C6C6C3E007E0000- <1>     db  000h, 03ch, 06ch, 06ch, 03eh, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
645 00016D79 00000000000000      <1>
646 00016D80 00386C6C38007C0000- <1>     db  000h, 038h, 06ch, 06ch, 038h, 000h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
646 00016D89 00000000000000      <1>
647 00016D90 0000303000303060C0- <1>     db  000h, 000h, 030h, 030h, 000h, 030h, 030h, 060h, 0c0h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
647 00016D99 C6C67C00000000      <1>
648 00016DA0 000000000000FEC0C0- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h, 000h
648 00016DA9 C0C00000000000      <1>
649 00016DB0 000000000000FE0606- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0feh, 006h, 006h, 006h, 006h, 000h, 000h, 000h, 000h, 000h
649 00016DB9 06060000000000      <1>
650 00016DC0 00C0C0C2C6CC183060- <1>     db  000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 060h, 0ceh, 09bh, 006h, 00ch, 01fh, 000h, 000h
650 00016DC9 CE9B060C1F0000      <1>
651 00016DD0 00C0C0C2C6CC183066- <1>     db  000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 066h, 0ceh, 096h, 03eh, 006h, 006h, 000h, 000h
651 00016DD9 CE963E06060000      <1>
652 00016DE0 0000181800181818183C- <1>     db  000h, 000h, 018h, 018h, 000h, 018h, 018h, 018h, 03ch, 03ch, 03ch, 018h, 000h, 000h, 000h, 000h
652 00016DE9 3C3C1800000000      <1>
653 00016DF0 0000000000366CD86C- <1>     db  000h, 000h, 000h, 000h, 000h, 036h, 06ch, 0d8h, 06ch, 036h, 000h, 000h, 000h, 000h, 000h, 000h
653 00016DF9 36000000000000      <1>
654 00016E00 0000000000D86C366C- <1>     db  000h, 000h, 000h, 000h, 000h, 0d8h, 06ch, 036h, 06ch, 0d8h, 000h, 000h, 000h, 000h, 000h, 000h
654 00016E09 D8000000000000      <1>
655 00016E10 114411441144114411- <1>     db  011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h
655 00016E19 44114411441144      <1>
656 00016E20 55AA55AA55AA55AA55- <1>     db  055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah
656 00016E29 AA55AA55AA55AA      <1>
657 00016E30 DD77DD77DD77DD77DD- <1>     db  0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h
657 00016E39 77DD77DD77DD77      <1>
658 00016E40 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
658 00016E49 18181818181818      <1>
659 00016E50 18181818181818F818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
659 00016E59 18181818181818      <1>
660 00016E60 1818181818F818F818- <1>     db  018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
660 00016E69 18181818181818      <1>
661 00016E70 36363636363636F636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
661 00016E79 36363636363636      <1>
662 00016E80 00000000000000FE36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
662 00016E89 36363636363636      <1>
663 00016E90 0000000000F818F818- <1>     db  000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
663 00016E99 18181818181818      <1>
664 00016EA0 3636363636F606F636- <1>     db  036h, 036h, 036h, 036h, 036h, 0f6h, 006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
664 00016EA9 36363636363636      <1>
665 00016EB0 3636363636363636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
665 00016EB9 36363636363636      <1>
666 00016EC0 0000000000FE06F636- <1>     db  000h, 000h, 000h, 000h, 000h, 0feh, 006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
666 00016EC9 36363636363636      <1>
667 00016ED0 3636363636F606FE00- <1>     db  036h, 036h, 036h, 036h, 036h, 0f6h, 006h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
667 00016ED9 00000000000000      <1>
668 00016EE0 36363636363636FE00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
668 00016EE9 00000000000000      <1>
669 00016EF0 1818181818F818F800- <1>     db  018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
669 00016EF9 00000000000000      <1>
```

```
670 00016F00 00000000000000F818- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
670 00016F09 18181818181818      <1>
671 00016F10 1818181818181F00- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
671 00016F19 00000000000000      <1>
672 00016F20 18181818181818FF00- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
672 00016F29 00000000000000      <1>
673 00016F30 00000000000000FF18- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
673 00016F39 18181818181818      <1>
674 00016F40 1818181818181F18- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
674 00016F49 18181818181818      <1>
675 00016F50 00000000000000FF00- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
675 00016F59 00000000000000      <1>
676 00016F60 18181818181818FF18- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
676 00016F69 18181818181818      <1>
677 00016F70 18181818181F181F18- <1>     db  018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
677 00016F79 18181818181818      <1>
678 00016F80 363636363636363736- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
678 00016F89 36363636363636      <1>
679 00016F90 363636363637303F00- <1>     db  036h, 036h, 036h, 036h, 036h, 037h, 030h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
679 00016F99 00000000000000      <1>
680 00016FA0 00000000003F303736- <1>     db  000h, 000h, 000h, 000h, 000h, 03fh, 030h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
680 00016FA9 36363636363636      <1>
681 00016FB0 3636363636F700FF00- <1>     db  036h, 036h, 036h, 036h, 036h, 0f7h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
681 00016FB9 00000000000000      <1>
682 00016FC0 0000000000FF00F736- <1>     db  000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0f7h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
682 00016FC9 36363636363636      <1>
683 00016FD0 363636363637303736- <1>     db  036h, 036h, 036h, 036h, 036h, 037h, 030h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
683 00016FD9 36363636363636      <1>
684 00016FE0 0000000000FF00FF00- <1>     db  000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
684 00016FE9 00000000000000      <1>
685 00016FF0 3636363636F700F736- <1>     db  036h, 036h, 036h, 036h, 036h, 0f7h, 000h, 0f7h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
685 00016FF9 36363636363636      <1>
686 00017000 1818181818FF00FF00- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
686 00017009 00000000000000      <1>
687 00017010 363636363636FF00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
687 00017019 00000000000000      <1>
688 00017020 0000000000FF00FF18- <1>     db  000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
688 00017029 18181818181818      <1>
689 00017030 00000000000000FF36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
689 00017039 36363636363636      <1>
690 00017040 363636363636363F00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
690 00017049 00000000000000      <1>
691 00017050 18181818181F181F00- <1>     db  018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
691 00017059 00000000000000      <1>
692 00017060 00000000001F181F18- <1>     db  000h, 000h, 000h, 000h, 000h, 01fh, 018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
692 00017069 18181818181818      <1>
693 00017070 000000000000003F36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 03fh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
693 00017079 36363636363636      <1>
694 00017080 363636363636FF36- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
694 00017089 36363636363636      <1>
695 00017090 1818181818FF18FF18- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
695 00017099 18181818181818      <1>
696 000170A0 18181818181818F800- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
696 000170A9 00000000000000      <1>
697 000170B0 000000000000001F18- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
697 000170B9 18181818181818      <1>
698 000170C0 FFFFFFFFFFFFFFFF- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
698 000170C9 FFFFFFFFFFFFFF      <1>
699 000170D0 000000000000FFFF- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
699 000170D9 FFFFFFFFFFFFFF      <1>
700 000170E0 F0F0F0F0F0F0F0F0- <1>     db  0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h
700 000170E9 F0F0F0F0F0F0F0      <1>
701 000170F0 0F0F0F0F0F0F0F0F0F- <1>     db  00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh
701 000170F9 0F0F0F0F0F0F0F      <1>
702 00017100 FFFFFFFFFFFFF0000- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
702 00017109 00000000000000      <1>
703 00017110 000000000076DCD8D8- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0dch, 0d8h, 0d8h, 0d8h, 0dch, 076h, 000h, 000h, 000h, 000h
703 00017119 D8DC7600000000      <1>
704 00017120 000078CCCCCCD8CCC6- <1>     db  000h, 000h, 078h, 0cch, 0cch, 0cch, 0d8h, 0cch, 0c6h, 0c6h, 0c6h, 0cch, 000h, 000h, 000h, 000h
704 00017129 C6C6CC00000000      <1>
705 00017130 0000FEC6C6C0C0C0C0- <1>     db  000h, 000h, 0feh, 0c6h, 0c6h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h
705 00017139 C0C0C000000000      <1>
706 00017140 00000000FE6C6C6C6C- <1>     db  000h, 000h, 000h, 000h, 0feh, 06ch, 06ch, 06ch, 06ch, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 000h
706 00017149 6C6C6C00000000      <1>
707 00017150 000000FEC660301830- <1>     db  000h, 000h, 000h, 0feh, 0c6h, 060h, 030h, 018h, 030h, 060h, 0c6h, 0feh, 000h, 000h, 000h, 000h
707 00017159 60C6FE00000000      <1>
708 00017160 00000000007ED8D8D8- <1>     db  000h, 000h, 000h, 000h, 000h, 07eh, 0d8h, 0d8h, 0d8h, 0d8h, 0d8h, 070h, 000h, 000h, 000h, 000h
708 00017169 D8D87000000000      <1>
709 00017170 000000006666666666- <1>     db  000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h, 066h, 07ch, 060h, 060h, 0c0h, 000h, 000h, 000h
709 00017179 7C6060C0000000      <1>
710 00017180 0000000076DC181818- <1>     db  000h, 000h, 000h, 000h, 076h, 0dch, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h
710 00017189 18181800000000      <1>
711 00017190 0000007E183C666666- <1>     db  000h, 000h, 000h, 07eh, 018h, 03ch, 066h, 066h, 066h, 03ch, 018h, 07eh, 000h, 000h, 000h, 000h
711 00017199 3C187E00000000      <1>
712 000171A0 000000386CC6C6FEC6- <1>     db  000h, 000h, 000h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h, 000h
712 000171A9 C66C3800000000      <1>
713 000171B0 0000386CC6C6C66C6C- <1>     db  000h, 000h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 06ch, 06ch, 06ch, 06ch, 0eeh, 000h, 000h, 000h, 000h
713 000171B9 6C6CEE00000000      <1>
714 000171C0 00001E30180C3E6666- <1>     db  000h, 000h, 01eh, 030h, 018h, 00ch, 03eh, 066h, 066h, 066h, 066h, 03ch, 000h, 000h, 000h, 000h
714 000171C9 66663C00000000      <1>
715 000171D0 00000000007EDBDBDB- <1>     db  000h, 000h, 000h, 000h, 000h, 07eh, 0dbh, 0dbh, 0dbh, 07eh, 000h, 000h, 000h, 000h, 000h, 000h
715 000171D9 7E000000000000      <1>
716 000171E0 00000003067EDBDBF3- <1>     db  000h, 000h, 000h, 003h, 006h, 07eh, 0dbh, 0dbh, 0f3h, 07eh, 060h, 0c0h, 000h, 000h, 000h, 000h
716 000171E9 7E60C000000000      <1>
717 000171F0 00001C3060607C6060- <1>     db  000h, 000h, 01ch, 030h, 060h, 060h, 07ch, 060h, 060h, 060h, 030h, 01ch, 000h, 000h, 000h, 000h
717 000171F9 60301C00000000      <1>
718 00017200 0000007CC6C6C6C6C6- <1>     db  000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
718 00017209 C6C6C600000000      <1>
719 00017210 00000000FE0000FE00- <1>     db  000h, 000h, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 000h, 000h, 000h
719 00017219 00FE0000000000      <1>
720 00017220 0000000018187E1818- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 07eh, 018h, 018h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h
720 00017229 0000FF00000000      <1>
721 00017230 00000030180C060C18- <1>     db  000h, 000h, 000h, 030h, 018h, 00ch, 006h, 00ch, 018h, 030h, 000h, 07eh, 000h, 000h, 000h, 000h
721 00017239 30007E00000000      <1>
722 00017240 0000000C1830603018- <1>     db  000h, 000h, 000h, 00ch, 018h, 030h, 060h, 030h, 018h, 00ch, 000h, 07eh, 000h, 000h, 000h, 000h
722 00017249 0C007E00000000      <1>
723 00017250 00000E1B1B18181818- <1>     db  000h, 000h, 00eh, 01bh, 01bh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
723 00017259 18181818181818      <1>
724 00017260 18181818181818D8- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0d8h, 0d8h, 0d8h, 070h, 000h, 000h, 000h, 000h
724 00017269 D8D87000000000      <1>
725 00017270 000000001818007E00- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 07eh, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h
725 00017279 18180000000000      <1>
726 00017280 000000000076DC0076- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h
726 00017289 DC000000000000      <1>
727 00017290 00386C6C3800000000- <1>     db  000h, 038h, 06ch, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
727 00017299 00000000000000      <1>
728 000172A0 0000000000000001818- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
728 000172A9 00000000000000      <1>
729 000172B0 0000000000000000018- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
729 000172B9 00000000000000      <1>
730 000172C0 000F0C0C0C0C0CEC6C- <1>     db  000h, 00fh, 00ch, 00ch, 00ch, 00ch, 00ch, 0ech, 06ch, 06ch, 03ch, 01ch, 000h, 000h, 000h, 000h
730 000172C9 6C3C1C00000000      <1>
731 000172D0 00D86C6C6C6C6C0000- <1>     db  000h, 0d8h, 06ch, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
731 000172D9 00000000000000      <1>
```

```
732 000172E0 0070D83060C8F80000- <1>     db  000h, 070h, 0d8h, 030h, 060h, 0c8h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
732 000172E9 00000000000000      <1>
733 000172F0 000000007C7C7C7C7C- <1>     db  000h, 000h, 000h, 000h, 07ch, 07ch, 07ch, 07ch, 07ch, 07ch, 07ch, 000h, 000h, 000h, 000h, 000h
733 000172F9 7C7C0000000000      <1>
734 00017300 00000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
734 00017309 00000000000000      <1>
735                              <1>
736                              <1> ; 01/01/2021 (TRDOS 386 v2.0.3)
737                              <1>
738                              <1> %if 0
739                              <1>
740                              <1> vgafont14alt:
741                              <1>     db  01dh, 000h, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h, 000h, 000h, 000h, 022h
742                              <1>     db  000h, 063h, 063h, 063h, 022h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 02bh, 000h
743                              <1>     db  000h, 000h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 02dh, 000h, 000h
744                              <1>     db  000h, 000h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 04dh, 000h, 000h, 0c3h
745                              <1>     db  0e7h, 0ffh, 0dbh, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 000h, 000h, 000h, 054h, 000h, 000h, 0ffh, 0dbh
746                              <1>     db  099h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 056h, 000h, 000h, 0c3h, 0c3h, 0c3h
747                              <1>     db  0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h, 057h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h
748                              <1>     db  0dbh, 0dbh, 0ffh, 066h, 066h, 000h, 000h, 000h, 058h, 000h, 000h, 0c3h, 0c3h, 066h, 03ch, 018h
749                              <1>     db  03ch, 066h, 0c3h, 0c3h, 000h, 000h, 000h, 059h, 000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h
750                              <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 05ah, 000h, 000h, 0ffh, 0c3h, 086h, 00ch, 018h, 030h, 061h
751                              <1>     db  0c3h, 0ffh, 000h, 000h, 000h, 06dh, 000h, 000h, 000h, 000h, 000h, 0e6h, 0ffh, 0dbh, 0dbh, 0dbh
752                              <1>     db  0dbh, 000h, 000h, 000h, 076h, 000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h
753                              <1>     db  000h, 000h, 000h, 077h, 000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 000h
754                              <1>     db  000h, 000h, 091h, 000h, 000h, 000h, 000h, 06eh, 03bh, 01bh, 07eh, 0d8h, 0dch, 077h, 000h, 000h
755                              <1>     db  000h, 09bh, 000h, 018h, 018h, 07eh, 0c3h, 0c0h, 0c0h, 0c3h, 07eh, 018h, 018h, 000h, 000h, 000h
756                              <1>     db  09dh, 000h, 000h, 0c3h, 066h, 03ch, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 000h, 000h, 000h, 09eh
757                              <1>     db  000h, 0fch, 066h, 066h, 07ch, 062h, 066h, 06fh, 066h, 066h, 0f3h, 000h, 000h, 000h, 0f1h, 000h
758                              <1>     db  000h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h, 000h, 0ffh, 000h, 000h, 000h, 0f6h, 000h, 000h
759                              <1>     db  018h, 018h, 000h, 000h, 0ffh, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h
760                              <1> vgafont16alt:
761                              <1>     db  01dh, 000h, 000h, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h, 000h, 000h, 000h
762                              <1>     db  000h, 030h, 000h, 000h, 03ch, 066h, 0c3h, 0c3h, 0dbh, 0dbh, 0c3h, 0c3h, 066h, 03ch, 000h, 000h
763                              <1>     db  000h, 000h, 04dh, 000h, 000h, 0c3h, 0e7h, 0ffh, 0ffh, 0dbh, 0c3h, 0c3h, 0c3h, 0c3h, 000h, 000h
764                              <1>     db  000h, 000h, 000h, 054h, 000h, 000h, 0ffh, 0dbh, 099h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch
765                              <1>     db  000h, 000h, 000h, 000h, 056h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch
766                              <1>     db  018h, 000h, 000h, 000h, 000h, 057h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh
767                              <1>     db  066h, 066h, 000h, 000h, 000h, 000h, 058h, 000h, 000h, 0c3h, 0c3h, 066h, 03ch, 018h, 018h, 03ch
768                              <1>     db  066h, 0c3h, 0c3h, 000h, 000h, 000h, 059h, 000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h
769                              <1>     db  018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 05ah, 000h, 000h, 0ffh, 0c3h, 086h, 00ch, 018h
770                              <1>     db  030h, 060h, 0c1h, 0c3h, 0ffh, 000h, 000h, 000h, 000h, 06dh, 000h, 000h, 000h, 000h, 000h, 0e6h
771                              <1>     db  0ffh, 0dbh, 0dbh, 0dbh, 0dbh, 0dbh, 000h, 000h, 000h, 000h, 076h, 000h, 000h, 000h, 000h, 000h
772                              <1>     db  0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h, 077h, 000h, 000h, 000h, 000h, 000h
773                              <1>     db  000h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 000h, 000h, 000h, 078h, 000h, 000h, 000h, 000h
774                              <1>     db  000h, 000h, 0c3h, 066h, 03ch, 018h, 03ch, 066h, 0c3h, 000h, 000h, 000h, 091h, 000h, 000h, 000h
775                              <1>     db  000h, 000h, 000h, 06eh, 03bh, 01bh, 07eh, 0d8h, 0dch, 077h, 000h, 000h, 000h, 09bh, 000h, 000h
776                              <1>     db  018h, 018h, 07eh, 0c3h, 0c0h, 0c0h, 0c3h, 07eh, 018h, 018h, 000h, 000h, 000h, 09dh, 000h, 000h
777                              <1>     db  000h, 000h, 0c3h, 066h, 03ch, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 000h, 000h, 000h, 000h
778                              <1>     db  09eh, 000h, 0fch, 066h, 066h, 07ch, 062h, 066h, 06fh, 066h, 066h, 066h, 0f3h, 000h, 000h, 000h
779                              <1>     db  000h, 0abh, 000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 060h, 0ceh, 09bh, 006h, 00ch, 01fh
780                              <1>     db  000h, 000h, 0ach, 000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 066h, 0ceh, 096h, 03eh, 006h
781                              <1>     db  006h, 000h, 000h, 000h
782                              <1>
783                              <1> %endif
3790
3791                                 ; 20/11/2020
3792                                 vbe2_bochs_vbios:
3793                                 ;   db "BOCHS/QEMU"
3794 00017310 424F4348532F51454D-        db "BOCHS/QEMU/VIRTUALBOX"  ; 26/11/2020
3794 00017319 552F5649525455414C-
3794 00017322 424F58
3795                                 ;vbe_vnumber equ vbe2_bochs_vbios + 28 ; "3" or "2"
3796                                 ; 26/11/2020
3797                                 vbe_vnumber equ vbe2_bochs_vbios + 30 ; "3" or "2"
3798                                 ; 15/11/2020
3799                                 vesa_vbe3_bios_msg:
3800                                     ;db " VESA VBE version 3 Video BIOS ..."
3801 00017325 205645534120564245-        db " VESA VBE3 Video BIOS ..." ; 26/11/2020
3801 0001732E 3320566964656F2042-
3801 00017337 494F53202E2E2E
3802 0001733E 0D0A0D0A00                db 0Dh, 0Ah, 0Dh, 0Ah, 0
3803
3804                                 ; 28/02/2021
3805 00017343 18                     truecolor: db 24
3806
3807                                 align 2
3808
3809                                 ; EPOCH Variables
3810                                 ; 13/04/2015 - Retro UNIX 386 v1 Beginning
3811                                 ; 09/04/2013 epoch variables
3812                                 ; Retro UNIX 8086 v1 Prototype: UNIXCOPY.ASM, 10/03/2013
3813                                 ;
3814 00017344 B207                   year:       dw 1970
3815 00017346 0100                   month:      dw 1
3816 00017348 0100                   day:        dw 1
3817 0001734A 0000                   hour:       dw 0
3818 0001734C 0000                   minute: dw 0
3819 0001734E 0000                   second: dw 0
3820
3821                                 DMonth:
3822 00017350 0000                       dw 0
3823 00017352 1F00                       dw 31
3824 00017354 3B00                       dw 59
3825 00017356 5A00                       dw 90
3826 00017358 7800                       dw 120
3827 0001735A 9700                       dw 151
3828 0001735C B500                       dw 181
3829 0001735E D400                       dw 212
3830 00017360 F300                       dw 243
3831 00017362 1101                       dw 273
3832 00017364 3001                       dw 304
3833 00017366 4E01                       dw 334
3834
3835                                 ; 15/11/2020
3836 00017368 00                     db  0
3837                                 kernel_version_msg: ; 17/04/2021
3838                                 ;;;;;;db "TRDOS (386) Kernel v2.0.4 by Erdogan Tan"
3839                                 ;;;;;db     "TRDOS (386) Kernel v2.0.5 by Erdogan Tan" ; 11/08/2022
3840                                 ;;;;db      "TRDOS (386) Kernel v2.0.6 by Erdogan Tan" ; 29/08/2023
3841                                 ;;;db       "TRDOS (386) Kernel v2.0.7 by Erdogan Tan" ; 20/10/2023
3842                                 ;;db"TRDOS (386) Kernel v2.0.8 by Erdogan Tan" ; 16/05/2024
3843                                 ;db "TRDOS (386) Kernel v2.0.9 by Erdogan Tan" ; 20/08/2024
3844 00017369 5452444F5320283338-     db  "TRDOS (386) Kernel v2.0.10 by Erdogan Tan" ; 11/01/2025
3844 00017372 3629204B65726E656C-
3844 0001737B 2076322E302E313020-
3844 00017384 6279204572646F6761-
3844 0001738D 6E2054616E
3845 00017392 00                     db  0
3846
3847                                 ; 20/02/2017
3848                                 KERNELFSIZE  equ $  ; 04/07/2016
3849
3850                                 bss_start:
```

```
                                    ABSOLUTE bss_start
                                    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
00017398 <res 280h>                     resb   80*8 ; INT 0 to INT 4Fh

                                    idt_end:

                                    ;alignb 4

                                    task_state_segment:
                                        ; 24/03/2015
00017618 ????                       tss.link:   resw 1
0001761A ????                               resw 1
                                    ; tss offset 4
0001761C ????????                   tss.esp0:   resd 1
00017620 ????                       tss.ss0:    resw 1
00017622 ????                               resw 1
00017624 ????????                   tss.esp1:   resd 1
00017628 ????                       tss.ss1:    resw 1
0001762A ????                               resw 1
0001762C ????????                   tss.esp2:   resd 1
00017630 ????                       tss.ss2:    resw 1
00017632 ????                               resw 1
                                    ; tss offset 28
00017634 ????????                   tss.CR3:    resd 1
00017638 ????????                   tss.eip:    resd 1
0001763C ????????                   tss.eflags: resd 1
                                    ; tss offset 40
00017640 ????????                   tss.eax:    resd 1
00017644 ????????                   tss.ecx:    resd 1
00017648 ????????                   tss.edx:    resd 1
0001764C ????????                   tss.ebx:    resd 1
00017650 ????????                   tss.esp:    resd 1
00017654 ????????                   tss.ebp:    resd 1
00017658 ????????                   tss.esi:    resd 1
0001765C ????????                   tss.edi:    resd 1
                                    ; tss offset 72
00017660 ????                       tss.ES:     resw 1
00017662 ????                               resw 1
00017664 ????                       tss.CS:         resw 1
00017666 ????                               resw 1
00017668 ????                       tss.SS:         resw 1
0001766A ????                               resw 1
0001766C ????                       tss.DS:         resw 1
0001766E ????                               resw 1
00017670 ????                       tss.FS:         resw 1
00017672 ????                               resw 1
00017674 ????                       tss.GS:         resw 1
00017676 ????                               resw 1
00017678 ????                       tss.LDTR:   resw 1
0001767A ????                               resw 1
                                    ; tss offset 100
0001767C ????                               resw 1
0001767E ????                       tss.IOPB:   resw 1
                                    ; tss offset 104
                                    tss_end:

00017680 ????????                   k_page_dir:  resd 1 ; Kernel's (System) Page Directory address
                                                        ; (Physical address = Virtual address)
00017684 ????????                   memory_size: resd 1 ; memory size in pages
00017688 ????????                   free_pages:  resd 1 ; number of free pages
0001768C ????????                   next_page:   resd 1 ; offset value in M.A.T. for
                                                        ; first free page search
00017690 ????????                   last_page:   resd 1 ; offset value in M.A.T. which
                                                        ; next free page search will be
                                                        ; stopped after it. (end of M.A.T.)
00017694 ????????                   first_page:  resd 1 ; offset value in M.A.T. which
                                                        ; first free page search
                                                        ; will be started on it. (for user)
00017698 ????????                   mat_size:    resd 1 ; Memory Allocation Table size in pages

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

                                    ; 02/09/2014 (Retro UNIX 386 v1)
                                    ; 04/12/2013 (Retro UNIX 8086 v1)
0001769C ????                       CRT_START:   resw 1         ; starting address in regen buffer
                                                        ; NOTE: active page only
0001769E <res 10h>                  CURSOR_POSN: resw 8 ; cursor positions for video pages
                                    ACTIVE_PAGE:
000176AE ??                         ptty:           resb 1 ; current tty
                                    ; 01/07/2015 - 29/01/2016
000176AF ??                         ccolor:          resb 1 ; current color attribute
                                    ; 26/10/2015
                                    ; 07/09/2014
000176B0 <res 14h>                  ttychr:      resw ntty+2 ; Character buffer (multiscreen)

                                    ; 18/05/2015 (03/06/2013 - Retro UNIX 8086 v1 feature only!)
000176C4 ????????                   p_time:      resd 1     ; present time (for systime & sysmdate)

                                    ; 18/05/2015 (16/08/2013 - Retro UNIX 8086 v1 feature only !)
                                    ; (open mode locks for pseudo TTYs)
                                    ; [ major tty locks (return error in any conflicts) ]
000176C8 <res 14h>                  ttyl:        resw ntty+2 ; opening locks for TTYs.

                                    ; 15/04/2015 (Retro UNIX 386 v1)
                                    ; 22/09/2013 (Retro UNIX 8086 v1)
000176DC <res Ah>                   wlist:       resb ntty+2 ; wait channel list (0 to 9 for TTYs)
                                    ; 15/04/2015 (Retro UNIX 386 v1)
                                    ;; 12/07/2014 -> sp_init set comm. parameters as 0E3h
                                    ;; 0 means serial port is not available
                                    ;;comprm: ; 25/06/2014
000176E6 ??                         com1p:       resb 1  ;;0E3h
000176E7 ??                         com2p:       resb 1  ;;0E3h

                                    ; 17/11/2015
                                    ; request for response (from the terminal)
000176E8 ????                       req_resp:    resw 1
                                    ; 07/11/2015
000176EA ??                         ccomport:    resb 1 ; current COM (serial) port
                                                   ; (0= COM1, 1= COM2)
                                    ; 09/11/2015
000176EB ??                         comqr:          resb 1 ; 'query or response' sign (u9.s, 'sndc')
```

```
3975                                    ; 07/11/2015
3976 000176EC ????                      rchar:          resw 1 ; last received char for COM 1 and COM 2
3977 000176EE ????                      schar:          resw 1 ; last sent char for COM 1 and COM 2
3978
3979                                    ; 22/08/2014 (RTC)
3980                                    ; (Packed BCD)
3981 000176F0 ??                        time_seconds: resb 1
3982 000176F1 ??                        time_minutes: resb 1
3983 000176F2 ??                        time_hours:   resb 1
3984 000176F3 ??                        date_wday:    resb 1
3985 000176F4 ??                        date_day:     resb 1
3986 000176F5 ??                        date_month:   resb 1
3987 000176F6 ??                        date_year:    resb 1
3988 000176F7 ??                        date_century: resb 1
3989
3990                                    ; 24/01/2016
3991 000176F8 ????????                  RTC_LH:           resd 1
3992 000176FC ??                        RTC_WAIT_FLAG: resb 1
3993 000176FD ??                        USER_FLAG:     resb 1
3994                                    ; 19/05/2016
3995                                    ;RTC_second:
3996 000176FE ??                        RTC_2Hz:       resb 1 ;  from 2Hz interrupt to 1Hz timer event function
3997
3998                                    %include 'diskbss.s'      ; UNINITIALIZED DISK (BIOS) DATA
   1                                <1> ; ****************************************************************************
   2                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.5 - diskbss.s
   3                                <1> ; ----------------------------------------------------------------------------
   4                                <1> ; Last Update: 06/08/2022 (Previous: 24/01/2016)
   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> ; 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> ;   (Unitialized Disk Parameters Data section for 'DISKIO.INC')
  22                                <1>
  23 000176FF ??                    <1> alignb 2
  24                                <1>
  25                                <1> ;----------------------------------------
  26                                <1> ;   TIMER DATA AREA               :
  27                                <1> ;----------------------------------------
  28                                <1>
  29                                <1> TIMER_LH:   ; 16/02/2015
  30 00017700 ????                  <1> TIMER_LOW:      resw      1               ; LOW WORD OF TIMER COUNT
  31 00017702 ????                  <1> TIMER_HIGH:     resw      1               ; HIGH WORD OF TIMER COUNT
  32 00017704 ??                    <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 00017705 ??                    <1> SEEK_STATUS:       resb   1
  39 00017706 ??                    <1> MOTOR_STATUS:      resb   1
  40 00017707 ??                    <1> MOTOR_COUNT:       resb   1
  41 00017708 ??                    <1> DSKETTE_STATUS:    resb   1
  42 00017709 ??????????????        <1> NEC_STATUS: resb   7
  43                                <1>
  44                                <1> ;----------------------------------------
  45                                <1> ;   ADDITIONAL MEDIA DATA         :
  46                                <1> ;----------------------------------------
  47                                <1>
  48 00017710 ??                    <1> LASTRATE:   resb   1
  49 00017711 ??                    <1> HF_STATUS:  resb   1
  50                                <1> ; 06/08/2022
  51                                <1> ;HF_ERROR:  resb   1
  52 00017712 ??                    <1> HF_INT_FLAG:       resb   1
  53                                <1> ; 06/08/2022
  54                                <1> ;HF_CNTRL:  resb   1
  55                                <1> ;DSK_STATE: resb   4
  56                                <1> ; 06/08/2022
  57 00017713 ????                  <1> DSK_STATE:  resb   2
  58 00017715 ????                  <1> DSK_TRK:    resb   2
  59                                <1>
  60                                <1> ;----------------------------------------
  61                                <1> ;   FIXED DISK DATA AREAS         :
  62                                <1> ;----------------------------------------
  63                                <1>
  64 00017717 ??                    <1> DISK_STATUS1:      resb   1             ; FIXED DISK STATUS
  65 00017718 ??                    <1> HF_NUM:            resb   1             ; COUNT OF FIXED DISK DRIVES
  66 00017719 ??                    <1> CONTROL_BYTE:      resb   1             ; HEAD CONTROL BYTE
  67                                <1> ;@PORT_OFF  resb   1             ; RESERVED (PORT OFFSET)
  68                                <1> ;port1_off  resb   1             ; Hard disk controller 1 - port offset
  69                                <1> ;port2_off  resb   1             ; Hard disk controller 2 - port offset
  70                                <1>
  71 0001771A ????                  <1> alignb 4
  72                                <1>
  73                                <1> ;HF_TBL_VEC:       resd   1             ; Primary master disk param. tbl. pointer
  74                                <1> ;HF1_TBL_VEC:      resd   1             ; Primary slave disk param. tbl. pointer
  75                                <1> HF_TBL_VEC: ; 22/12/2014
  76 0001771C ????????              <1> HDPM_TBL_VEC:      resd   1             ; Primary master disk param. tbl. pointer
  77 00017720 ????????              <1> HDPS_TBL_VEC:      resd   1             ; Primary slave disk param. tbl. pointer
  78 00017724 ????????              <1> HDSM_TBL_VEC:      resd   1             ; Secondary master disk param. tbl. pointer
  79 00017728 ????????              <1> HDSS_TBL_VEC:      resd   1             ; Secondary slave disk param. tbl. pointer
  80                                <1>
  81                                <1> ; 03/01/2015
  82 0001772C ??                    <1> LBAMode:           resb   1
  83                                <1>
  84                                <1> ; ****************************************************************************
3999
4000                                    ;;; Real Mode Data (10/07/2015 - BSS)
4001
4002                                    ;alignb 2
4003
4004                                    ; 10/01/2016
4005                                    %include 'trdoskx.s'      ; UNINITIALIZED KERNEL (Logical Drive & FS) DATA
   1                                <1> ; ****************************************************************************
   2                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.10) - UNINITIALIZED DATA : trdoskx.s
   3                                <1> ; ----------------------------------------------------------------------------
   4                                <1> ; Last Update: 19/12/2025 (Previous: 01/09/2024 - Kernel v2.0.9)
   5                                <1> ; ----------------------------------------------------------------------------
   6                                <1> ; Beginning: 04/01/2016
   7                                <1> ; ----------------------------------------------------------------------------
   8                                <1> ; Assembler: NASM version 2.11 (trdos386.s)
   9                                <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>
0001772D ??????             <1> alignb 4
                            <1>
                            <1> ; MAINPROG.ASM
00017730 ????????           <1> MainProgCfg_FileSize:   resd 1 ; 14/04/2016
00017734 ????????           <1> MainProgCfg_LineOffset: resd 1 ; 14/04/2016
                            <1>
00017738 ????????           <1> Current_VolSerial: resd 1
                            <1>
0001773C ????????           <1> Current_Dir_FCluster: resd 1
                            <1>
00017740 ??                 <1> Current_Dir_Level: resb 1
00017741 ??                 <1> Current_FATType: resb 1
00017742 ??                 <1> Current_Drv: resb 1
00017743 ??                 <1> Current_Dir_Drv:   resb 1 ; '?'
00017744 ??                 <1>                    resb 1 ; ':'
00017745 ??                 <1> Current_Dir_Root:  resb 1 ; '/'
00017746 <res 5Ah>          <1> Current_Directory: resb 90
000177A0 ??                 <1> End_Of_Current_Dir_Str: resb 1
000177A1 ??                 <1> Current_Dir_StrLen: resb 1
                            <1>
000177A2 ??                 <1> CursorColumn:      resb 1
000177A3 ??                 <1> CmdArgStart:    resb 1
                            <1>
                            <1> ; 03/02/2016
000177A4 <res 4Eh>          <1> Remark:            resb 78
                            <1>
000177F2 <res 50h>          <1> CommandBuffer:     resb 80
                            <1>
                            <1> ;TextBuffer:       ;resb 256
                            <1> ; 04/07/2025
00017842 <res 80h>          <1> TextBuffer: resb 128
000178C2 <res 80h>          <1> RunPathBuffer:     resb 128
                            <1>
                            <1> MasterBootBuff:
00017942 <res 1BEh>         <1> MasterBootCode: resb 1BEh
00017B00 <res 40h>          <1> PartitionTable: resb 64
00017B40 ????               <1> MBIDCode: resw 1
                            <1>
                            <1> PTable_Buffer:
00017B42 <res 40h>          <1> PTable_hd0: resb 64
00017B82 <res 40h>          <1> PTable_hd1: resb 64
00017BC2 <res 40h>          <1> PTable_hd2: resb 64
00017C02 <res 40h>          <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> ; 22/05/2024
00017C42 ????????           <1> HD_LBA_yes: resd 1
                            <1> ; 13/08/2020
00017C46 ??                 <1> scount: resb 1 ; 16/05/2016 (diskio.s, 'int33h:')
                            <1> ; 22/05/2024
                            <1>     ;resb 1
                            <1>     ;resb 1
                            <1>     ;resb 1
                            <1> ;HD_LBA_yes: resb 1
00017C47 ??                 <1> PP_Counter: resb 1
00017C48 ??                 <1> EP_Counter: resb 1
                            <1> ; 13/08/2020
00017C49 ??                 <1> LD_Counter: resb 1
                            <1>
                            <1> ; 30/08/2020
00017C4A ????????           <1> MBR_EP_StartSector: resd 1
                            <1>             ; Extd partition start sector as in MBR
00017C4E ????????           <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
00017C52 <res 200h>         <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:
00017E52 ????????           <1> FAT_CurrentCluster: resd 1
00017E56 ??                 <1> FAT_BuffValidData: resb 1
00017E57 ??                 <1> FAT_BuffDrvName: resb 1
00017E58 ????               <1> FAT_BuffOffset: resw 1
00017E5A ????????           <1> FAT_BuffSector: resd 1
                            <1>
00017E5E ????????           <1> FAT_ClusterCounter: resd 1
00017E62 ????????           <1> LastCluster: resd 1
                            <1>
                            <1> ; 16/05/2016
                            <1> ;; 18/03/2016 (TRDOS v2.0)
                            <1> ;ClusterBuffer_Valid: resb 1
                            <1>
                            <1> ; 29/07/2022
                            <1> ;resb 1
                            <1> ; 02/12/2023
00017E66 ??                 <1> P_TIMER: resb 1 ; diskette change check (2 seconds)
                            <1>
                            <1> Dir_BuffDescriptor:
00017E67 ??                 <1> DirBuff_DRV: resb 1
00017E68 ??                 <1> DirBuff_FATType: resb 1
00017E69 ??                 <1> DirBuff_ValidData: resb 1
00017E6A ????               <1> DirBuff_CurrentEntry: resw 1
00017E6C ????               <1> DirBuff_LastEntry: resw 1
00017E6E ????????           <1> DirBuff_Cluster: resd 1
00017E72 ????               <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>
```

```
00017E74 ????????            <1> FreeClusterCount: resd 1
                             <1>
00017E78 ????????            <1> VolSize_Unit1:   resd 1
00017E7C ????????            <1> VolSize_Unit2:   resd 1
                             <1>
00017E80 ????????            <1> Vol_Tot_Sec_Str_Start:        resd 1
00017E84 <res Ah>            <1> Vol_Tot_Sec_Str:       resb 10
00017E8E ??                  <1> Vol_Tot_Sec_Str_End:          resb 1
00017E8F ??                  <1> resb 1
00017E90 ????????            <1> Vol_Free_Sectors_Str_Start: resd 1
00017E94 <res Ah>            <1> Vol_Free_Sectors_Str:         resb 10
00017E9E ??                  <1> Vol_Free_Sectors_Str_End:   resb 1
                             <1>
                             <1> ; 10/02/2016
00017E9F ??                  <1> RUN_CDRV: resb 1 ; CMD_INTR.ASM  ; 09/11/2011
                             <1>
                             <1> ; 24/01/2016
00017EA0 <res 80h>           <1> PATH_Array:     resb 128 ; DIR.ASM ; 09/10/2011
                             <1> ; 06/02/2016
00017F20 ????????            <1> CCD_DriveDT:       resd 1 ; DIR.ASM ; (word)
00017F24 ??                  <1> CCD_Level:  resb 1 ; DIR.ASM
00017F25 ??                  <1> Last_Dir_Level:    resb 1 ; DIR.ASM
                             <1> ;
00017F26 ????                <1> CDLF_AttributesMask: resw 1 ; DIR.ASM
00017F28 ????????            <1> CDLF_FNAddress:    resd 1 ; DIR.ASM (word)
00017F2C ????                <1> CDLF_DEType:       resw 1 ; DIR.ASM
                             <1> ;
00017F2E ??                  <1> CD_COMMAND: resb 1 ; DIR.ASM
                             <1>
00017F2F ??                  <1> alignb 4
                             <1>
                             <1> ; 29/01/2016
00017F30 ??                  <1> Program_Exit:      resb 1 ; CMD_INTR.ASM  ; 09/11/2011
                             <1>
                             <1> ;alignb 4
                             <1> ; 23/02/2016
00017F31 ??                  <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
00017F32 ??                  <1> retry_count:       resb 1 ; DISK_IO.ASM ; 20/07/2011 (CHS_RetryCount)
00017F33 ??                  <1> disk_rw_err:       resb 1 ; DISK_IO.ASM ; (Disk_IO_err_code)
00017F34 ????????            <1> sector_count:      resd 1 ; DISK_IO.ASM ; (Disk_RW_SectorCount)
                             <1>
                             <1> ; 06/02/2016 (long name)
00017F38 ????                <1> FDE_AttrMask:         resw 1 ; DIR.ASM
00017F3A ????                <1> AmbiguousFileName: resw 1 ; DIR.ASM
00017F3C ??                  <1> PreviousAttr:         resb 1 ; DIR.ASM
                             <1> ;
00017F3D ??                  <1> LongNameFound:   resb 1     ; DIR.ASM
00017F3E ??                  <1> LFN_EntryLength: resb 1   ; DIR.ASM
00017F3F ??                  <1> LFN_CheckSum:    resb 1   ; DIR.ASM
00017F40 <res 84h>           <1> LongFileName:    resb 132 ; DIR.ASM
                             <1>
                             <1> ;PATH_Array_Ptr: resw 1 ; DIR.ASM
00017FC4 ??                  <1> PATH_CDLevel:       resb 1 ; DIR.ASM
00017FC5 ??                  <1> PATH_Level:  resb 1 ; DIR.ASM
                             <1>
                             <1> ; 07/02/2016
00017FC6 <res Dh>            <1> Dir_File_Name:     resb 13 ; DIR.ASM ; 09/10/2011
                             <1>
                             <1> ; 10/02/2016
00017FD3 <res Dh>            <1> Dir_Entry_Name:    resb 13 ; DIR.ASM
                             <1>
                             <1> alignb 2
                             <1>
00017FE0 ????                <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
00017FE2 ??                  <1> FindFile_Drv:                resb 1
00017FE3 <res 41h>           <1> FindFile_Directory:          resb 65
00018024 <res Dh>            <1> FindFile_Name:               resb 13
                             <1> FindFile_LongNameEntryLength:
00018031 ??                  <1> FindFile_LongNameYes:        resb 1 ; Sign for longname procedures
                             <1> ;Above 80 bytes form
                             <1> ;TR-DOS Source/Destination File FullName Format/Structure
00018032 ????                <1> FindFile_AttributesMask:  resw 1
00018034 <res 20h>           <1> FindFile_DirEntry:   resb 32
00018054 ????????            <1> FindFile_DirFirstCluster: resd 1
00018058 ????????            <1> FindFile_DirCluster:         resd 1
0001805C ????                <1> FindFile_DirEntryNumber:  resw 1
0001805E ????                <1> FindFile_MatchCounter:       resw 1
00018060 ????                <1> FindFile_Reserved:   resw 1 ; 06/03/2016
                             <1>
00018062 ????????            <1> First_Path_Pos: resd 1     ; DIR.ASM ; 09/10/2011
00018066 ????????            <1> Last_Slash_Pos: resd 1     ; DIR.ASM
                             <1>
                             <1> ; 10/02/2016
0001806A ????                <1> File_Count:      resw 1    ; DIR.ASM ; 09/10/2011
0001806C ????                <1> Dir_Count:       resw 1
0001806E ????????            <1> Total_FSize:     resd 1
00018072 ????????            <1> TFS_Dec_Begin:   resd 1
00018076 <res Ah>            <1>                  resb 10
00018080 ??                  <1> TFS_Dec_End:     resb 1
                             <1>
00018081 ??                  <1> PrintDir_RowCounter: resb 1
                             <1>
00018082 ????                <1> alignb 4
                             <1> ; 15/02/2015 ('show' command variables)
00018084 ????????            <1> Show_FDT:   resd 1
00018088 ????????            <1> Show_LDDDT: resd 1
0001808C ????????            <1> Show_Cluster:      resd 1
00018090 ????????            <1> Show_FileSize:     resd 1
00018094 ????????            <1> Show_FilePointer: resd 1
00018098 ????                <1> Show_ClusterPointer: resw 1
0001809A ????                <1> Show_ClusterSize: resw 1
0001809C ??                  <1> Show_RowCount:     resb 1
                             <1>
0001809D ??????              <1> alignb 4
                             <1> ; 21/02/2016
000180A0 ????????            <1> DelFile_FNPointer: resd 1 ; ; CMD_INTR.ASM (word) ; 09/11/2011
                             <1> ; 27/02/2016
                             <1> ; DIR.ASM (09/10/2011)
000180A4 ????????            <1> DelFile_FCluster:  resd 1
000180A8 ????                <1> DelFile_EntryCounter:     resw 1
000180AA ??                  <1> DelFile_LNEL:             resb 1
                             <1> ; 17/07/2025
                             <1> ;resb 1
                             <1> ; 22/02/2016
000180AB ??                  <1> UPDLMDT_CDirLevel: resb 1
000180AC ????????            <1> UPDLMDT_CDirFCluster:     resd 1
                             <1>
                             <1> ; DIR.ASM
```

```
258 000180B0 ????????           <1> mkdir_DirName_Offset:     resd 1
259 000180B4 ????????           <1> mkdir_FFCluster:   resd 1
260 000180B8 ????????           <1> mkdir_LastDirCluster:     resd 1
261 000180BC ????????           <1> mkdir_FreeSectors: resd 1
262 000180C0 ????               <1> mkdir_attrib:             resw 1
263 000180C2 ??                 <1> mkdir_SecPerClust: resb 1
264 000180C3 ??                 <1> mkdir_add_new_cluster:    resb 1
265 000180C4 <res Dh>           <1> mkdir_Name:        resb 13
266 000180D1 ????               <1> resw 1 ; 01/03/2016
267                             <1> ; 27/02/2016
268 000180D3 ??                 <1> RmDir_MultiClusters:      resb 1
269 000180D4 ????????           <1> RmDir_DirEntryOffset:     resd 1 ; 01/03/2016 (word -> dword)
270 000180D8 ????????           <1> RmDir_ParentDirCluster: resd 1
271 000180DC ????????           <1> RmDir_DirLastCluster:   resd 1
272 000180E0 ????????           <1> RmDir_PreviousCluster:  resd 1
273                             <1>
274                             <1> ; 17/07/2025 - TRDOS 386 v2.0.10
275 000180E4 ????????           <1> rmdir_dir_fcluster:       resd 1
276 000180E8 ??                 <1> rmdir_drv:         resb 1
277                             <1>
278 000180E9 ??????             <1> alignb 4
279                             <1> ; DRV_FAT.ASM ; 21/08/2011
280 000180EC ????????           <1> gffc_next_free_cluster:  resd 1
281 000180F0 ????????           <1> gffc_first_free_cluster: resd 1
282 000180F4 ????????           <1> gffc_last_free_cluster:  resd 1
283                             <1>
284                             <1> ;29/04/2016
285                             <1> Cluster_Index: ; resd 1
286                             <1> ; 22/02/2016
287 000180F8 ????????           <1> ClusterValue:      resd 1
288                             <1> ; 04/03/2016
289 000180FC ??                 <1> Attributes: resb 1
290                             <1> ;;CFS_error:  resb 1 ;; 01/03/2016
291 000180FD ??                 <1> resb 1
292 000180FE ??                 <1> CFS_OPType: resb 1
293 000180FF ??                 <1> CFS_Drv:    resb 1
294 00018100 ????????           <1> CFS_CC:         resd 1
295 00018104 ????????           <1> CFS_FAT32FSINFOSEC: resd 1
296 00018108 ????????           <1> CFS_FAT32FC: resd 1
297                             <1>
298                             <1> ; 27/02/2016
299                             <1> ;alignb 4
300 0001810C ????????           <1> glc_prevcluster: resd 1 ; DRV_FAT.ASM (21/08/2011)
301                             <1> ; 22/10/2016
302 00018110 ????????           <1> glc_index:   resd 1 ; Last Cluster Index (22/10/2016)
303                             <1>
304                             <1> ; DIR.ASM
305 00018114 ????               <1> DLN_EntryNumber: resw 1
306 00018116 ??                 <1> DLN_40h:     resb 1
307                             <1> ; 28/02/2016
308 00018117 ??                 <1> TCC_FATErr:  resb 1 ; DRV_FAT.ASM
309                             <1>
310                             <1> alignb 4
311                             <1> ; DIR.ASM (09/10/2011)
312 00018118 ????               <1> LCDE_EntryIndex: resw 1 ; LCDE_EntryOffset
313 0001811A ????               <1> LCDE_ClusterSN:  resw 1
314 0001811C ????????           <1> LCDE_Cluster:       resd 1
315 00018120 ????????           <1> LCDE_ByteOffset: resd 1
316                             <1>
317                             <1> ;alignb4
318                             <1> ; 06/03/2016 (word -> dword)
319                             <1> ; CMD_INTR.ASM (01/08/2010)
320 00018124 ????????           <1> SourceFilePath:         resd 1
321 00018128 ????????           <1> DestinationFilePath: resd 1
322                             <1>
323                             <1> ;alignb 4
324                             <1> ; 06/03/2016
325                             <1> ; FILE.ASM (09/10/2011)
326                             <1> ;Source File Structure (same with 'Find File' Structure)
327 0001812C ??                 <1> SourceFile_Drv:                   resb 1
328 0001812D <res 41h>          <1> SourceFile_Directory:             resb 65
329 0001816E <res Dh>           <1> SourceFile_Name:          resb 13
330                             <1> SourceFile_LongNameEntryLength:
331 0001817B ??                 <1> SourceFile_LongNameYes:           resb 1 ; Sign for longname procedures
332                             <1> ;Above 80 bytes
333                             <1> ;is TR-DOS Source File FullName Format/Structure
334 0001817C ????               <1> SourceFile_AttributesMask: resw 1
335 0001817E <res 20h>          <1> SourceFile_DirEntry:              resb 32
336 0001819E ????????           <1> SourceFile_DirFirstCluster:       resd 1
337 000181A2 ????????           <1> SourceFile_DirCluster:            resd 1
338 000181A6 ????               <1> SourceFile_DirEntryNumber: resw 1
339 000181A8 ????               <1> SourceFile_MatchCounter:  resw 1
340                             <1> ; 16/03/2016
341 000181AA ??                 <1> SourceFile_SecPerClust:           resb 1
342 000181AB ??                 <1> SourceFile_Reserved:              resb 1
343                             <1> ; Above is 128 bytes
344                             <1>
345                             <1> ;Destination File Structure (same with 'Find File' Structure)
346 000181AC ??                 <1> DestinationFile_Drv:              resb 1
347 000181AD <res 41h>          <1> DestinationFile_Directory:        resb 65
348 000181EE <res Dh>           <1> DestinationFile_Name:             resb 13
349                             <1> DestinationFile_LongNameEntryLength:
350 000181FB ??                 <1> DestinationFile_LongNameYes:      resb 1 ; Sign for longname procedures
351                             <1> ;Above 80 bytes
352                             <1> ;is TR-DOS Destination File FullName Format/Structure
353 000181FC ????               <1> DestinationFile_AttributesMask: resw 1
354 000181FE <res 20h>          <1> DestinationFile_DirEntry: resb 32
355 0001821E ????????           <1> DestinationFile_DirFirstCluster: resd 1
356 00018222 ????????           <1> DestinationFile_DirCluster:       resd 1
357 00018226 ????               <1> DestinationFile_DirEntryNumber: resw 1
358 00018228 ????               <1> DestinationFile_MatchCounter:     resw 1
359                             <1> ; 16/03/2016
360 0001822A ??                 <1> DestinationFile_SecPerClust:      resb 1
361 0001822B ??                 <1> DestinationFile_Reserved: resb 1
362                             <1> ; Above is 128 bytes
363                             <1>
364                             <1> ; 24/04/2016
365 0001822C ????               <1> resw 1
366                             <1>
367                             <1> ; 10/03/2016
368                             <1> ; FILE.ASM
369 0001822E ??                 <1> move_cmd_phase:       resb 1
370 0001822F ??                 <1> msftdf_sf_df_drv:  resb 1
371 00018230 ????????           <1> msftdf_drv_offset: resd 1
372                             <1>
373                             <1> ; 11/03/2016
374                             <1> ; DRV_FAT.ASM (21/08/2011)
375 00018234 ????????           <1> FAT_anc_LCluster:  resd 1
376 00018238 ????????           <1> FAT_anc_FFCluster: resd 1
377                             <1>
378                             <1> ;alignb 4
379                             <1>
380                             <1> ; 14/03/2016
381                             <1> ; TRDOS 386 = TRDOS v2.0 feature only !
```

```
382                             <1> ; 'allocate_memory_block' in 'memory.s'
383 0001823C ????????           <1> mem_ipg_count:     resd 1 ; page count (for contiguous allocation)
384 00018240 ????????           <1> mem_pg_count:      resd 1 ; page count (for count down)
385 00018244 ????????           <1> mem_aperture:      resd 1 ; contiguous free pages (current)
386 00018248 ????????           <1> mem_max_aperture: resd 1 ; maximum value of contiguous free pages
387 0001824C ????????           <1> mem_pg_pos: resd 1 ; mem. position (page #) of current aperture
388 00018250 ????????           <1> mem_max_pg_pos: resd 1 ; mem. position (page #) of max. aperture
389                             <1>
390                             <1> ; 15/03/2016
391                             <1> ; FILE.ASM ('copy_source_file_to_destination_file')
392 00018254 ??                 <1> copy_cmd_phase:       resb 1
393 00018255 ??                 <1> csftdf_rw_err:           resb 1
394 00018256 ??                 <1> DestinationFileFound: resb 1
395 00018257 ??                 <1> csftdf_cdrv:             resb 1
396 00018258 ????????           <1> csftdf_filesize:      resd 1
397                             <1> ; TRDOS386 (TRDOS v2.0)
398                             <1> MainProgCfg_mem_addr: ; 19/12/2025
399 0001825C ????????           <1> csftdf_sf_mem_addr:   resd 1
400                             <1> MainProgCfg_mem_bsize: ; 19/12/2025
401 00018260 ????????           <1> csftdf_sf_mem_bsize:  resd 1
402                             <1> ;
403                             <1> MainProgCfg_cluster: ; 19/12/2025
404 00018264 ????????           <1> csftdf_sf_cluster:    resd 1 ; 16/03/2016
405 00018268 ????????           <1> csftdf_df_cluster:    resd 1
406                             <1> MainProgCfg_r_size: ; 19/12/2025
407                             <1> ; 16/03/2016
408 0001826C ????????           <1> csftdf_r_size:        resd 1
409 00018270 ????????           <1> csftdf_w_size:        resd 1
410                             <1> MainProgCfg_rbytes: ; 19/12/2025
411 00018274 ????????           <1> csftdf_sf_rbytes:     resd 1
412 00018278 ????????           <1> csftdf_df_wbytes:     resd 1
413 0001827C ??                 <1> csftdf_percentage:    resb 1
414                             <1> ; 17/03/2016
415 0001827D ??                 <1> csftdf_videopage:     resb 1
416 0001827E ????               <1> csftdf_cursorpos:     resw 1
417 00018280 ????????           <1> csftdf_sf_drv_dt:     resd 1
418 00018284 ????????           <1> csftdf_df_drv_dt:     resd 1
419                             <1> ; 01/09/2024
420                             <1> ; 29/08/2024
421                             <1> ;csftdf_df_dclust:    resd 1
422                             <1> ;csftdf_df_dindex:    resd 1
423                             <1>
424                             <1> ; 21/03/2016
425                             <1> ; 20/03/2016
426                             <1> ; FILE.ASM
427 00018288 ????????           <1> createfile_Name_Offset:  resd 1
428 0001828C ????????           <1> createfile_FreeSectors:  resd 1
429 00018290 ????????           <1> createfile_size:         resd 1
430 00018294 ????????           <1> createfile_FFCluster:    resd 1 ; 11/03/2016
431 00018298 ????????           <1> createfile_LastDirCluster: resd 1
432 0001829C ????????           <1> createfile_Cluster:      resd 1
433 000182A0 ????????           <1> createfile_PCluster:     resd 1
434 000182A4 ??                 <1> createfile_attrib:  resb 1
435 000182A5 ??                 <1> createfile_SecPerClust:  resb 1
436 000182A6 ????               <1> createfile_DirIndex:     resw 1
437 000182A8 ????????           <1> createfile_CCount:  resd 1
438                             <1> ; 19/12/2025
439                             <1> ;createfile_BytesPerSec: resw 1 ; 23/03/2016
440 000182AC ??                 <1> createfile_wfc:             resb 1
441 000182AD ??                 <1> createfile_UpdatePDir:     resb 1 ; 31/03/2016
442                             <1>
443                             <1> ;alignb 4
444                             <1>
445                             <1> ; 11/04/2016
446 000182AE ????               <1> env_var_length:    resw 1
447                             <1>
448                             <1> alignb 4
449                             <1>
450                             <1> ; 25/04/2016
451 000182B0 ??                 <1> readi.valid:       resb 1 ; valid data (>0 = valid for readi)
452 000182B1 ??                 <1> readi.drv:  resb 1 ; drive number (0, 1,2,3,4..)
453 000182B2 ??                 <1> readi.spc:  resb 1 ; sectors per cluster for 'readi' drive
454 000182B3 ??                 <1> readi.s_index:  resb 1 ; sector index in current cluster (buffer)
455 000182B4 ????????           <1> readi.sector:      resd 1 ; current disk sector
456 000182B8 ????               <1> readi.bpc:  resw 1 ; bytes per cluster - 1
457 000182BA ????               <1> readi.offset:      resw 1 ; byte offset in cluster buffer
458 000182BC ????????           <1> readi.cluster:  resd 1 ; current cluster number
459 000182C0 ????????           <1> readi.c_index:     resd 1 ; cluster index of the current cluster (0,1,2,3..)
460 000182C4 ????????           <1> readi.fclust:      resd 1 ; first cluster of the current cluster
461                             <1> ; 19/12/2025
462                             <1> ;readi.fs_index: resd 1 ; sector index in disk/file section (for Singlix FS)
463                             <1> ;readi.buffer:     resd 1 ; readi sector buffer address
464                             <1>
465                             <1> ;alignb 4
466                             <1>
467 000182C8 ??                 <1> writei.valid:      resb 1 ; valid data (>0 = valid for writei)
468 000182C9 ??                 <1> writei.drv: resb 1 ; drive number (0, 1,2,3,4..)
469 000182CA ??                 <1> writei.spc: resb 1 ; sectors per cluster for 'writei' drive
470 000182CB ??                 <1> writei.s_index: resb 1 ; sector index in current cluster (buffer)
471 000182CC ????????           <1> writei.sector:     resd 1 ; current disk sector
472 000182D0 ????               <1> writei.bpc: resw 1 ; bytes per cluster - 1
473 000182D2 ????               <1> writei.offset:     resw 1 ; byte offset in cluster buffer
474 000182D4 ????????           <1> writei.cluster: resd 1 ; current cluster number
475 000182D8 ????????           <1> writei.c_index:    resd 1 ; cluster index of the current cluster (0,1,2,3..)
476 000182DC ????????           <1> writei.fclust:  resd 1 ; first cluster of the current cluster
477 000182E0 ????????           <1> writei.fs_index: resd 1 ; sector index in disk/file section (for Singlix FS)
478                             <1> ;writei.buffer:    resd 1 ; writei sector buffer address
479 000182E4 ????????           <1> writei.lclust:     resd 1 ; writei last cluster (mget_w) ; 23/10/2016
480 000182E8 ????????           <1> writei.l_index:    resd 1 ; writei last cluster index (mget_w) ; 23/10/2016
481 000182EC ??                 <1> writei.ofn: resb 1 ; open file number (to be written) ; 23/10/2016
482                             <1>
483 000182ED ??????             <1> alignb 4
484                             <1>
485                             <1> ; 29/04/2016
486 000182F0 ????????           <1> Run_CDirFC: resd 1
487                             <1> ;Run_Auto_Path:    resb 1
488                             <1> ; 03/07/2025
489                             <1> Run_Path_Length:
490 000182F4 ????????           <1>                 resd 1
491 000182F8 ????????           <1> Run_Auto_Path:     resb 1
492                             <1> Run_Manual_Path:
493 000182FC ??                 <1>                 resb 1 ; 0 -> auto path sequence needed
494 000182FD ??                 <1> EXE_ID:            resb 1
495 000182FE ??                 <1> EXE_dot:    resb 1
496 000182FF ??                 <1>                 resb 1 ; 03/07/2025
497                             <1>
498                             <1> ; 06/05/2016
499 00018300 ????????           <1> mainprog_return_addr: resd 1
500 00018304 ????????           <1> last_error: resd 1  ; this will be used to return error code to MainProg
501                             <1>                     ; 'lasterror' keyword will be used later to get the
502                             <1>                     ; last error code/number/status.
503                             <1> ; 12/05/2016
504 00018308 ????????           <1> video_eax:  resd 1  ; eax return value of video function
505                             <1>
```

```
                                <1> ; 01/06/2016
0001830C ????????               <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])
00018310 ??                     <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')
00018311 ??                     <1> p_change:   resb 1  ; process change status (for timer events)
                                <1> ; 23/05/2016 - TRDOS 386 ('clock')
00018312 ??                     <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
00018313 ??                     <1> timer_events:      resb 1  ; number of (active) timer events, <= 16
                                <1>
                                <1> ; 24/06/2016
00018314 ??                     <1> w_str_cmd:  resb 1 ; WRITE_STRING command (0,1,2,3) ; video.s
00018315 ??                     <1> p_crt_mode: resb 1  ; previous video mode (=3 or 0), backup mark/sign
                                <1> ; 26/06/2016
00018316 ??                     <1> p_crt_page: resb 1  ; previous active page (for 'set_mode')
                                <1> ; 04/07/2016
00018317 ??                     <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
00018318 ????                   <1> CRT_LEN:    resw 1  ; length of regen buffer in bytes
0001831A <res 10h>              <1> cursor_pposn:      resw 8  ; cursor positions backup
                                <1>
                                <1> ; 10/07/2016 ('VGA_FONT_SETUP', INT 43H address for x86 real mode bios)
0001832A ????????               <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
0001832E ??                     <1> VGA_MTYPE:  resb 1  ; 0=CTEXT,1=MTEXT,2=CGA,3=PLANAR1,4=PLANAR4,5=LINEAR
                                <1>
                                <1> ; 23/10/2016
0001832F ??                     <1> setfmod:    resb 1  ; update last modification date&time sign (if >0)
                                <1>                     ; (it is Open File Number + 1, if > 0)
                                <1> ; 27/08/2024
00018330 ??                     <1> setfclust:  resb 1  ; first cluster of file
                                <1>                     ; (is used by update lmdt proc)
00018331 ??????                 <1> alignb 4
                                <1>
                                <1> ; 16/10/2016
00018334 ????????               <1> FFF_UBuffer:       resd 1  ; User's buffer address for FFF & FNF system calls
                                <1> ; 15/10/2016
00018338 ??                     <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)
00018339 ??                     <1> FFF_Attrib: resb 1  ; Find First File attributes for Find Next File (LB)
0001833A ??                     <1> FFF_RType:  resb 1  ; FFF return type (0 = Basic, >0 = complete) (HB)
                                <1> ; 16/10/2016 - 05/10/2016 (Set Working Path)
0001833B ??                     <1> SWP_inv_fname:     resb 1 ; Set Working Path - Invalid File Name
0001833C ????                   <1> SWP_Mode:   resw 1  ; Set Working Path - Mode
0001833E ??                     <1> SWP_DRV:    resb 1  ; Set Working Path - Drive
0001833F ??                     <1> SWP_DRV_chg:       resb 1  ; Set Working Path - Drive Change
                                <1>
                                <1> ; 27/02/2017
00018340 ??                     <1> fpready:    resb 1 ; '80387 fpu is ready' flag
                                <1>
                                <1> ; 17/04/2021
                                <1> ; (DEVICE parameters is disabled as temporary)
                                <1>
                                <1> ; 08/10/2016
                                <1> ;device_name:      resb 9  ; capitalized (and zero padded) device name
                                <1>                     ; (example: "TTY0",0,0,0,0,0")
                                <1>
00018341 ??????                 <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)
                                <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>
                                    <1> ;IDEV_NAME:  resb 8*NUMIDEV
                                    <1>                     ; 8 byte names of installable device drivers
                                    <1>
                                    <1> ;IDEV_TYPE:  resb NUMIDEV ; Driver type of installable device drivers
                                    <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.)
                                    <1> ;IDEV_OADDR: resd NUMIDEV ; open function addr for installable dev driver
                                    <1> ;IDEV_CADDR: resd NUMIDEV ; close function addr for installable dev driver
                                    <1> ;IDEV_RADDR: resd NUMIDEV ; read function addr for installable dev driver
                                    <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
                                    <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
                                    <1> ;DEV_R_OWNER:      resb NUMOFDEVICES    ; Reading owner no (u.uid) of devices
                                    <1> ;DEV_R_OPENCOUNT: resb NUMOFDEVICES  ; Reading open count
                                    <1> ;DEV_W_OWNER:      resb NUMOFDEVICES    ; Writing owner no (u.uid) of devices
                                    <1> ;DEV_W_OPENCOUNT: resb NUMOFDEVICES  ; Writing open count
                                    <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
                                    <1> ;DEV_OPENMODE:     resb NUMOFDEVICES    ; 1 = read mode
                                    <1>                                         ; 2 = write mode
                                    <1>                                         ; 3 = read & write
                                    <1>                                         ; 0 = not open (free)
                                    <1> ;DEV_NAME_PTR:     resd NUMOFDEVICES    ; pointers to name addresses of drivers
                                    <1>                                         ; Address base: KDEV_NAME+
                                    <1>                                         ; or IDEV_NAME+
                                    <1> ;DEV_R_POINTER:    resd NUMOFDEVICES    ; reading pointer, writing pointer
                                    <1> ;DEV_W_POINTER:    resd NUMOFDEVICES    ; sector number if block device
                                    <1>                                         ; character offset if char device
                                    <1> alignb 4
                                    <1>
                                    <1> ; 06/10/2016
                                    <1> ; Open File Parameters
00018344 <res 80h>                  <1> OF_FCLUSTER:       resd OPENFILES  ; First clusters of open files
000183C4 <res 20h>                  <1> OF_DRIVE:   resb OPENFILES  ; Logical DOS drive numbers of open files
000183E4 <res 20h>                  <1> OF_MODE:    resb OPENFILES  ; Open mode (1 = read, 2 = write, 3 = r&w)
00018404 <res 20h>                  <1> OF_STATUS:  resb OPENFILES  ; (bit 0 = read, bit 1 = write)
00018424 <res 20h>                  <1> OF_OPENCOUNT:      resb OPENFILES  ; Open counts of open files
00018444 <res 80h>                  <1> OF_POINTER: resd OPENFILES ; File seek/read/write pointer
000184C4 <res 80h>                  <1> OF_SIZE:    resd OPENFILES ; File sizes of open files (in bytes)
00018544 <res 80h>                  <1> OF_DIRFCLUSTER:    resd OPENFILES  ; Directory First Clusters of open files
000185C4 <res 80h>                  <1> OF_DIRCLUSTER:     resd OPENFILES  ; Directory (Entry) Clusters of open files
00018644 <res 80h>                  <1> OF_VOLUMEID:       resd OPENFILES  ; Vol ID for removable drives of open files
000186C4 <res 80h>                  <1> OF_CCLUSTER:       resd OPENFILES  ; Current clusters of open files
00018744 <res 80h>                  <1> OF_CCINDEX: resd OPENFILES  ; Cluster index numbers of current clusters
                                    <1> ; 24/10/2016
000187C4 <res 40h>                  <1> OF_DIRENTRY:       resw OPENFILES  ; Directory entry index no. in dir cluster
                                    <1>                                    ; Sector index = entry index / 16
                                    <1> ;alignb 2
                                    <1>
                                    <1> DTA:        ;resd 24       ; Find First File data transfer area
00018804 <res 18h>                  <1>             resb 24        ; 29/07/2022
                                    <1>
                                    <1> ; 19/12/2016
0001881C ??                         <1> tcallback:  resb 1         ; Timer callback method flag for 'systimer'
0001881D ??                         <1> trtc:               resb 1         ; Timer interrupt type flag for 'systimer'
                                    <1> ; 20/02/2017
0001881E ??                         <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)
0001881F ??                         <1> alignb 4
                                    <1> ; 13/04/2017
                                    <1> ;DEV_INTR:  resb NUMOFDEVICES ; Device Interrupt (IRQ) number + 1
                                    <1>                               ; (0= not available, 1= IRQ 0, 16= IRQ 15)
00018820 <res 40h>                  <1> DEV_INT_HNDLR:     resd 16        ; Device Interrupt Handler addr, if > 0
                                    <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
00018860 <res 9h>                   <1> IRQ.owner:  resb 9         ; owner, 0 = free, >0 = [u.uno]
00018869 <res 9h>                   <1> IRQ.dev:    resb 9         ; 0 = default/kernel, >0 = device number
00018872 <res 9h>                   <1> IRQ.method: resb 9         ; 0 = Signal Response Byte, 1 = Callback
0001887B <res 9h>                   <1> IRQ.srb:    resb 9         ; Signal Response/Return Byte value
00018884 <res 24h>                  <1> IRQ.addr:   resd 9         ; Rignal Response Byte address (physical)
                                    <1>                            ; or Callback service address (virtual)
                                    <1> ; 28/02/2017
000188A8 ????????                   <1> IRQ_cr3:    resd 1         ; for saving cr3 register in IRQ handler
000188AC ??                         <1> IRQnum:             resb 1         ; IRQ number for IRQ handler (trdosk8.s)
                                    <1>
                                    <1> ; 10/04/2017
                                    <1> ; 03/04/2017
                                    <1> ; UNINITIALIZED AUDIO DATA
000188AD ??????                     <1> alignb 4
000188B0 ??                         <1> audio_pci:  resb 1
000188B1 ??                         <1> audio_device:      resb 1
                                    <1> ;audio_mode:       resb 1
000188B2 ??                         <1> audio_intr: resb 1
000188B3 ??                         <1> audio_busy: resb 1  ; Busy flag for audio irq ; 21/04/2017
                                    <1> ;audio_reserved: resb 1
                                    <1> ; 20/11/2023
000188B4 ????                       <1> NAMBAR:             resw 1  ; Native Audio Mixer Base Address
                                    <1> NABMBAR:    ; 02/10/2023 (NABMBAR = audio_io_base)
000188B6 ????                       <1> audio_io_base:     resw 1  ; Base I/O address of audio device
000188B8 ????????                   <1> audio_dev_id:      resd 1  ; BUS/DEV/FN ; 00000000BBBBBBBBDDDDDFFF00000000
000188BC ????????                   <1> audio_vendor:      resd 1
000188C0 ????????                   <1> audio_stats_cmd: resd 1
                                    <1> ;
000188C4 ????????                   <1> audio_buffer:      resd 1  ; virtual address of user's audio buffer
000188C8 ????????                   <1> audio_p_buffer:    resd 1  ; Physical address of user's audio buffer
000188CC ????????                   <1> audio_buff_size: resd 1 ; user's audio buffer size (half buffer size)
000188D0 ????????                   <1> audio_dma_buff: resd 1  ; dma buffer address
000188D4 ????????                   <1> audio_dmabuff_size: resd 1 ; dma buffer size (2 * half buffer size)
                                    <1> ; 05/06/2024
000188D8 ????????                   <1> dma_hbuff_size: resd 1 ; dma half buffer size
                                    <1> ;
```

```
754 000188DC ??                 <1> audio_flag: resb 1  ; dma buffer flag (1st half = 0, 2nd half = 1)
755 000188DD ??                 <1> audio_user: resb 1  ; user number of the owner
756 000188DE ??                 <1> audio_cb_mode:      resb 1  ; 0 = signal response byte method
757                             <1>                     ; 1 = callback method
758                             <1>                     ; 2 = s.r.b. method with auto increment
759 000188DF ??                 <1> audio_srb:  resb 1  ; signal response byte value
760 000188E0 ????????           <1> audio_cb_addr:      resd 1  ; callback service address or s.r.b. address
761                             <1>                     ; (s.r.b. addr is physical, cbs addr is virtual)
762                             <1>
763 000188E4 ??                 <1> audio_bps:  resb 1  ; selected mode: 8 bit, 16 bit
764 000188E5 ??                 <1> audio_stmo: resb 1  ; selected mode: mono /stereo
765 000188E6 ????               <1> audio_freq:         resw 1  ; sampling rate
766                             <1> ; 20/11/2023
767 000188E8 ??                 <1> VRA:        resb 1
768 000188E9 ??                 <1> audio_mode: resb 1
769                             <1>
770                             <1> ; 21/04/2017
771 000188EA ??                 <1> audio_play_cmd: resb 1  ; Play/Stop command (1 = play, 0 = stop)
772                             <1> ; 21/11/2023
773                             <1> ;audio_civ: ; 28/05/2017 ; Current Buffer Index (AC'97)
774                             <1> ; 23/05/2024
775                             <1> LVI:                ; AC'97 Last Valid Buffer Index
776 000188EB ??                 <1> audio_flag_eol:     resb 1  ; End of Link status (vt8233, EOL/FLAG)
777                             <1>
778                             <1> audio_master_volume:
779 000188EC ??                 <1> audio_master_volume_l: resb 1 ; sound volume (lineout) left channel
780 000188ED ??                 <1> audio_master_volume_r: resb 1 ; sound volume (lineout) right channel
781                             <1>
782                             <1> ; 24/05/2024
783                             <1> audio_pcmo_volume:
784 000188EE ??                 <1> audio_pcmo_volume_l: resb 1 ; PCM out volume left channel
785 000188EF ??                 <1> audio_pcmo_volume_r: resb 1 ; PCM out volume right channel
786                             <1>
787                             <1> ; 02/06/2024
788                             <1> ;alignb 4
789                             <1>
790                             <1> ; 20/11/2023
791                             <1> ; 28/05/2017
792                             <1> ; AC'97 Audio Controller Base Adress Registers
793                             <1> ;NAMBAR:    resw 1  ; Native Audio Mixer Base Address
794                             <1> ;NABMBAR:   resw 1  ; Native Audio Bus Mastering Base Address
795                             <1>
796                             <1> ; 02/06/2024
797                             <1> alignb 8
798                             <1>
799                             <1> ; 21/04/2017
800 000188F0 <res 400h>          <1> audio_bdl_buff:     resd 32*8 ; VT8233 (AC97) BDL Buffer Size
801                             <1> ; 12/05/2017
802 00018CF0 ????????           <1> base_addr:  resd 1  ; 'direct_memory_access' (memory.s)
803                             <1>
804                             <1> ; 01/06/2024
805 00018CF4 ??                 <1> reset:              resb 1  ; AC97 init
806                             <1>
807                             <1> ; 28/08/2017
808                             <1> ; 20/08/2017
809                             <1>             ;resb 1 ;
810 00018CF5 ??                 <1> dma_user:   resb 1  ; user number for sysdma
811 00018CF6 ??                 <1> dma_channel:        resb 1  ; dma channel for sysdma
812 00018CF7 ??                 <1> dma_mode:   resb 1  ; dma mode for sysdma
813 00018CF8 ????????           <1> dma_addr:   resd 1  ; dma buffer physical addr for sysdma
814 00018CFC ????????           <1> dma_size:   resd 1  ; dma buffer size (in bytes) for sysdma
815 00018D00 ????????           <1> dma_start:  resd 1  ; dma start address for sysdma
816 00018D04 ????????           <1> dma_count:  resd 1  ; dma count (in bytes) for sysdma
817                             <1>
818                             <1> ; 04/12/2023
819                             <1> %if 0
820                             <1>
821                             <1> alignb 65536
822                             <1> ; 09/08/2017
823                             <1> ; 12/05/2017
824                             <1> sb16_dma_buffer: resb 65536 ; DMA buffer for sb16 audio playing.
825                             <1>
826                             <1> %endif
4006                                ; 24/01/2016
4007                                %include 'ubss.s'  ; UNINITIALIZED KERNEL (USER) DATA
   1                            <1> ; ****************************************************************************
   2                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.9) - UNINITIALIZED USER DATA : ubss.s
   3                            <1> ; ----------------------------------------------------------------------------
   4                            <1> ; Last Update: 19/08/2024  (Previous: 07/08/2022)
   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> ; 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> ; Ref: Retro UNIX 386 v1.2 Kernel (v0.2.2.3) - ux.s - 15/07/2022
  29                            <1>
  30                            <1> alignb 2
  31                            <1>
  32                            <1> %if 0
  33                            <1>
  34                            <1> inode:
  35                            <1>     ;; 11/03/2013.
  36                            <1>     ;;Derived from UNIX v1 source code 'inode' structure (ux).
  37                            <1>     ;;i.
  38                            <1>     ;
  39                            <1>     ;i.flgs: resw 1
  40                            <1>     ;i.nlks: resb 1
  41                            <1>     ;i.uid:  resb 1
  42                            <1>         ;i.size: resw 1 ; size
  43                            <1>     ;i.dskp: resw 8 ; 16 bytes
  44                            <1>     ;i.ctim: resd 1
  45                            <1>     ;i.mtim: resd 1
  46                            <1>     ;i.rsvd: resw 1 ; Reserved (ZERO/Undefined word for UNIX v1)
  47                            <1>
  48                            <1>     ; 26/01/2020
  49                            <1>     ; Retro UNIX 386 v2.0 - Modified UNIX v7 inode model
```

```
                        <1>     ;       (15/09/2029 .. 18/12/2019)
                        <1>
                        <1>     i.flgs:   resw 1      ; /* mode and type of file */
                        <1>     i.nlks:   resw 1      ; /* number of links to file */
                        <1>     i.uid:    resw 1      ; /* owner's user id */  - 0 to 65535 -
                        <1>     i.gid:    resb 1      ; /* owner's group id */ - o to 255 -
                        <1>     i.size_h: resb 1      ; /* number of bytes in file */ ; byte 5
                        <1>     i.size:   resd 1 ; size       ; /* number of bytes in file */
                        <1>     i.dskp:   resd 10 ; 40 bytes ; /* disk block addresses */
                        <1>     i.atim:   resd 1      ; /* time last accessed */
                        <1>     i.mtim:   resd 1      ; /* time last modified */
                        <1>     i.ctim:   resd 1      ; /* time created */
                        <1>
                        <1> I_SIZE      equ $ - inode
                        <1>
                        <1> %endif
                        <1>
                        <1> process:
                        <1>     ; 23/07/2022 - TRDOS 386 Kernel v2.0.5
                        <1>     ; 27/02/2022
                        <1>     ; 12/01/2022 - Retro UNIX 386 v1.2
                        <1>     ; 19/12/2016
                        <1>     ; 21/05/2016
                        <1>     ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
                        <1>     ; 06/05/2015 - Retro UNIX 386 v1
                        <1>     ; 11/03/2013 - 05/02/2014 (Retro UNIX 8086 v1)
                        <1>     ;Derived from UNIX v1 source code 'proc' structure (ux).
                        <1>     ;p.
                        <1>
00018D08 <res 20h>      <1>         p.pid:   resw nproc
00018D28 <res 20h>      <1>         p.ppid:  resw nproc
                        <1>     ;p.break: resw nproc ; 12/01/2022 (p.break is not used)
00018D48 <res 10h>      <1>         p.ttyc:  resb nproc ; console tty in Retro UNIX 8086 v1.
                        <1>     ; 27/02/2022 (p.waitc is not used)
                        <1>     ;p.waitc: resb nproc ; waiting channel in Retro UNIX 8086 v1.
00018D58 <res 10h>      <1>     p.link:  resb nproc
00018D68 <res 10h>      <1>     p.stat:  resb nproc
                        <1>
                        <1>     ; 06/05/2015 (Retro UNIX 386 v1 feature only !)
00018D78 <res 40h>      <1>     p.upage: resd nproc ; Physical address of the process's
                        <1>                         ; 'user' structure
                        <1>     ; 21/05/2016
                        <1>     ; 19/05/2016 (TRDOS 386 feature only!)
00018DB8 <res 10h>      <1>     p.timer: resb nproc ; number of timer events of the processs
                        <1>
                        <1>     ; 19/12/2016
00018DC8 <res 40h>      <1>     p.tcb:  resd nproc ; timer callback service address (if > 0)
                        <1>
                        <1>     ; 18/08/2024 (TRDOS 386 v2.0.9)
00018E08 <res 10h>      <1>     p.exitc: resb nproc ; exit code of the child
                        <1>
                        <1> P_SIZE      equ $ - process
                        <1>
                        <1> ; fsp table (original UNIX v1)
                        <1> ;
                        <1> ;Entry
                        <1> ;           15                                      0
                        <1> ;  1      |---|--------------------------------------|
                        <1> ;         |r/w|       i-number of open file          |
                        <1> ;         |---|--------------------------------------|
                        <1> ;         |               device number              |
                        <1> ;         |------------------------------------------|
                        <1> ;    (*)  | offset pointer, i.e., r/w pointer to file |
                        <1> ;         |------------------------------------------|
                        <1> ;         |  flag that says    | number of processes  |
                        <1> ;         |   file deleted     | that have file open  |
                        <1> ;         |------------------------------------------|
                        <1> ;  2      |                                          |
                        <1> ;         |------------------------------------------|
                        <1> ;         |                                          |
                        <1> ;         |------------------------------------------|
                        <1> ;         |                                          |
                        <1> ;         |------------------------------------------|
                        <1> ;         |                                          |
                        <1> ;  3      |                                          |
                        <1> ;         |                                          |
                        <1> ;
                        <1> ; (*) Retro UNIX 386 v1 modification: 32 bit offset pointer
                        <1>
                        <1> ; 27/03/2020 - Retro UNIX 386 v2 - FSP (OPEN FILES) TABLE
                        <1>
                        <1> ;Entry
                        <1> ;           15                    7                 0
                        <1> ;  1      |---------------------------------------|
                        <1> ;         |              i-number of open file          |
                        <1> ;         |---------------------------------------|
                        <1> ;         |        high word of 32 bit i-number       |
                        <1> ;         |---------------------------------------|
                        <1> ;         | open mode & status  |   device number     |
                        <1> ;         |---------------------------------------|
                        <1> ;         |    reserved byte    |     open count      |
                        <1> ;         |---------------------------------------|
                        <1> ;         | offset pointer, i.e., r/w pointer to file |
                        <1> ;         |---------------------------------------|
                        <1> ;         |   64 bit file offset pointer (bit 16-31)  |
                        <1> ;         |---------------------------------------|
                        <1> ;         |   64 bit file offset pointer (bit 32-47)  |
                        <1> ;         |---------------------------------------|
                        <1> ;         |   64 bit file offset pointer (bit 48-63)  |
                        <1> ;         |---------------------------------------|
                        <1> ;  2      |                                       |
                        <1> ;         |---------------------------------------|
                        <1> ;         |                                       |
                        <1> ;         |---------------------------------------|
                        <1> ;         |                                       |
                        <1> ;         |---------------------------------------|
                        <1> ;         |                                       |
                        <1> ;         |---------------------------------------|
                        <1> ;         |                                       |
                        <1>
                        <1> %if 0
                        <1>
                        <1> ; (Retro UNIX 386 v1.2 - ux.s - 15/07/2022)
                        <1> ; 22/11/2021
                        <1> ; 21/07/2021 - Retro UNIX 386 v2 open file structure revision
                        <1>
                        <1> struc file  ; open files (fsp) structure ; (*)
                        <1>   .inode:  resw 1  ; inode number of open file (32 bit)
                        <1>   .i32:       resw 1  ; higher word of inode number (reserved)
                        <1>   .drive:  resb 1  ; logical drive (disk) number
                        <1>   .flags:  resb 1  ; open mode and status
                        <1>   .count:  resb 1  ; number of processes that have file open
                        <1>   ;.rsvd:  resb 1  ; reserved byte (for next versions)
```

```
                            <1>   .mnt:    resb 1  ; mnttab index+1 (0 = not mounted)
                            <1>   .offset: resd 1  ; file offset/pointer (64 bit)
                            <1>   .o64:        resd 1  ; higher 32 bit of file offset
                            <1>  .size:  ; = 16
                            <1> endstruc
                            <1>
                            <1> %endif
                            <1>
                            <1> ; 23/07/2022
                            <1> ;fsp:      resb nfiles * 16 ; (*)
                            <1>
00018E18 ??                 <1> idev:      resb 1
00018E19 ??                 <1> cdev:      resb 1
                            <1>
                            <1> ; 15/04/2015
                            <1> ;fsp:      resb nfiles * 10 ; 11/05/2015 (8 -> 10)
                            <1> ;idev:     resw 1 ; device number is 1 byte in Retro UNIX 8086 v1 !
                            <1> ;cdev:     resw 1 ; device number is 1 byte in Retro UNIX 8086 v1 !
                            <1>
                            <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
                            <1>
00018E1A ??                 <1> rdev:      resb 1 ; root device number ; Retro UNIX 8086 v1 feature only!
                            <1>             ; as above, for physical drives numbers in following table
00018E1B ??                 <1> mdev:      resb 1 ; mounted device number ; Retro UNIX 8086 v1 feature only!
                            <1>
                            <1> ; 23/07/2022
                            <1> ;; 15/04/2015
                            <1> ;active: resb 1
                            <1> ;    resb 1 ; 09/06/2015
                            <1>
                            <1> ; 23/07/2022
                            <1> ;mnti:      resw 1
                            <1> ; 07/08/2022
00018E1C ????               <1> mpid:      resw 1
                            <1> ;rootdir: resw 1
00018E1E ????????           <1> rootdir: resd 1
                            <1>
                            <1> ; 21/05/2016 - TRDOS 386 (TRDOS v2.0) - priority levels, 3 run queues
                            <1> runq:
00018E22 ????               <1> runq_event:  resw 1 ; high priority, 'run for event'           ; 2
00018E24 ????               <1> runq_normal:       resw 1 ; normal/regular priority, 'run as regular' ; 1
00018E26 ????               <1> runq_background: resw 1 ; low priority, 'run on background'         ; 0
                            <1> ;
                            <1> ; 23/07/2022
                            <1> ;imod:     resb 1
                            <1> ;smod:     resb 1
                            <1> ;mmod:     resb 1
00018E28 ??                 <1> sysflg:    resb 1
00018E29 ??                 <1>     resb 1
                            <1>
00018E2A ????               <1> alignb 4
                            <1>
                            <1> user:
                            <1>     ; 23/07/2022 - TRDOS 386 Kernel v2.0.5
                            <1>     ; 04/12/2021 - Retro UNIX 386 v1.2
                            <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>
00018E2C ????????           <1>     u.sp:    resd 1 ; esp (kernel stack at the beginning of 'sysent')
00018E30 ????????           <1>     u.usp:   resd 1 ; esp (kernel stack points to user's registers)
00018E34 ????????           <1>     u.r0:    resd 1 ; eax
00018E38 ????               <1>     u.cdir:  resw 1
00018E3A ????               <1>              resw 1 ; 23/07/2022
00018E3C ??                 <1>     u.cdrv:  resb 1 ; 23/07/2022
00018E3D ??                 <1>              resb 1
00018E3E <res Ah>           <1>     u.fp:    resb 10
                            <1>     ;u.fp:   resb OPENFILES ; 23/07/202
                            <1>     u.fsp:   ; 23/07/2022
00018E48 ????????           <1>     u.fofp:  resd 1
00018E4C ????????           <1>     u.dirp:  resd 1
00018E50 ????????           <1>     u.namep:  resd 1
00018E54 ????????           <1>     u.off:   resd 1
                            <1>     ;        resd 1 ; 23/07/2022 (64 bit fptr)
00018E58 ????????           <1>     u.base:  resd 1
00018E5C ????????           <1>     u.count:  resd 1
00018E60 ????????           <1>     u.nread:  resd 1
00018E64 ????????           <1>     u.break:  resd 1 ; break
                            <1>     ; 10/01/2017 (TRDOS 386, relocation and dword alignment)
                            <1>     ; tty number (rtty, rcvt, wtty)
00018E68 ????               <1>     u.ttyp:  resw 1
00018E6A ??                 <1>     u.ttyn:  resb 1 ; 28/07/2013 - Retro Unix 8086 v1 feature only !
00018E6B ??                 <1>     u.mode:   resb 1 ; 23/07/2022
                            <1>     ;u.resb:  resb 1 ; 10/01/2017 (TRDOS 386, temporary)
00018E6C <res 10h>          <1>     u.dirbuf: resb 16 ; 04/12/2015 (10 -> 16)
                            <1>     ;u.pri:  resw 1 ; 14/02/2014
00018E7C ??                 <1>     u.quant:  resb 1 ; Retro UNIX 8086 v1 Feature only ! (uquant)
00018E7D ??                 <1>              resb 1 ; 23/07/2022
00018E7E ??                 <1>     u.pri:   resb 1 ; Modification: 21/05/2016 (priority levels: 0, 1, 2)
00018E7F ??                 <1>              resb 1 ; 23/07/2022
00018E80 ????               <1>     u.intr:  resw 1
00018E82 ????               <1>     u.quit:  resw 1
                            <1>     ;u.emt:  resw 1 ; 10/10/2013
                            <1>     ;u.ilgins: resw 1 ; 10/01/2017
                            <1>     ;u.cdrv:  resw 1 ; cdev
00018E84 ??                 <1>     u.bsys:  resb 1
00018E85 ??                 <1>     u.uno:   resb 1
                            <1>     ; 23/07/2022
                            <1>     ;u.uid:  resw 1 ; uid ; 27/03/2021 - Retro UNIX 386 v2
```

```
298                                     <1>     ;u.ruid:  resw 1      ; 16 bit uid
299                                     <1>     ;u.gid:   resb 1 ; gid        ; 27/03/2021 - Retro UNIX 386 v2
300                                     <1>     ;u.rgid:  resb 1
301                                     <1>     ; 23/07/2022
302                                     <1>     ;u.procp: resd 1 ; /* pointer to proc structure */
303 00018E86 ??                         <1>     u.uid:    resb 1 ; uid
304 00018E87 ??                         <1>     u.ruid:   resb 1
305 00018E88 ????????                   <1>         u.upage:  resd 1 ; 16/04/2015 - Retro Unix 386 v1 feature only !
306 00018E8C ????????                   <1>     u.pgdir:  resd 1 ; 09/03/2015 (page dir addr of process)
307 00018E90 ????????                   <1>     u.ppgdir: resd 1 ; 06/05/2015 (page dir addr of the parent process)
308 00018E94 ????????                   <1>     u.pbase:  resd 1 ; 20/05/2015 (physical base/transfer address)
309 00018E98 ????                       <1>     u.pcount: resw 1 ; 20/05/2015 (byte -transfer- count for page)
310                                     <1>     ;u.pncount: resw 1
311                                     <1>             ; 16/06/2015 (byte -transfer- count for page, 'namei', 'mkdir')
312                                     <1>     ;u.pnbase:  resd 1
313                                     <1>             ; 16/06/2015 (physical base/transfer address, 'namei', 'mkdir')
314                                     <1>                     ; 09/06/2015
315 00018E9A ??                         <1>     u.kcall:  resb 1 ; The caller is 'namei' (dskr) or 'mkdir' (dskw) sign
316 00018E9B ??                         <1>     u.brwdev: resb 1 ; Block device number for direct I/O (bread & bwrite)
317                                     <1>                      ; 24/07/2015 - 24/06/2015
318                                     <1>     ;u.args:  resd 1 ; arguments list (line) offset from start of [u.upage]
319                                     <1>                      ; (arg list/line is from offset [u.args] to 4096 in [u.upage])
320                                     <1>                      ; ([u.args] points to argument count -argc- address offset)
321                                     <1>                      ; 24/06/2015
322                                     <1>     ;u.core:  resd 1 ; physical start address of user's memory space (for sys exec)
323                                     <1>     ;u.ecore: resd 1 ; physical end address of user's memory space (for sys exec)
324                                     <1>     ;;;
325                                     <1>     ; 19/08/2024    ; TRDOS 386 v2.0.9 ; STDIN/STDOUT/STDERR
326 00018E9C ??                         <1>     u.stdin:  resb 1 ; 0 is tty-r or > 0 is file (redirection)
327 00018E9D ??                         <1>     u.stdout: resb 1 ; 0 is tty-w or > 0 is file (redirection)
328 00018E9E ??                         <1>     u.ungetc: resb 1 ; u.getc is valid if u.ungetc > 0
329 00018E9F ??                         <1>     u.getc:   resb 1 ; last char read on stdin
330                                     <1>     ;;;
331                                     <1>     ; last error number
332 00018EA0 ????????                   <1>     u.error:  resd 1 ; 28/07/2013 - 09/03/2015
333                                     <1>                      ; Retro UNIX 8086/386 v1 feature only!
334                                     <1>                     ; 21/09/2015 (debugging - page fault analyze)
335 00018EA4 ????????                   <1>     u.pfcount: resd 1 ; page fault count for (this) process (for sys geterr)
336                                     <1>             ; 19/12/2016 (TRDOS 386)
337 00018EA8 ????????                   <1>     u.tcb:    resd 1 ; Timer callback address/flag which will be used by timer int
338                                     <1>             ; 13/01/2017 (TRDOS 386)
339 00018EAC ??                         <1>     u.t_lock: resb 1 ; Timer interrupt (callback) lock (unlocked by 'sysrele')
340 00018EAD ??                         <1>     u.t_mode: resb 1 ; running mode during timer interrupt (0= system, 0FFh= user)
341                                     <1>             ; 26/02/2017 (TRDOS 386)
342 00018EAE ??                         <1>     u.irqc:   resb 1  ; Count of IRQ callback services (IRQs in use)
343                                     <1>             ; 28/02/2017 (TRDOS 386)
344 00018EAF ??                         <1>     u.irqwait: resb 1 ; IRQ waiting for callback service flag (IRQ number, If > 0)
345 00018EB0 ??                         <1>     u.r_lock: resb 1 ; 'IRQ callback service is in progress' flag (IRQ lock)
346 00018EB1 ??                         <1>     u.r_mode: resb 1 ; running mode during hadware interrupt
347                                     <1>     ; 23/07/2022
348 00018EB2 ??                         <1>     u.exit:   resb 1 ; exit code
349                                     <1>     ; 27/02/2017 (TRDOS 386)
350 00018EB3 ??                         <1>     u.fpsave: resb 1 ; TRDOS 386, 'save/restore FPU registers' flag
351                                     <1> alignb 4
352                                     <1>     ; !! wrong sizing in TRDOS 386 v2.0.4 (in 'ubss.s', 28/02/2017) !!
353                                     <1>     ;u.fpregs: resb 94 ; 94 byte area for saving and restoring FPU registers
354                                     <1>     ; 23/07/2022
355 00018EB4 <res 6Ch>                  <1>     u.fpregs: resb 108 ; 108 byte area for saving and restoring FPU registers
356                                     <1>
357                                     <1> alignb 4
358                                     <1>
359                                     <1> U_SIZE      equ $ - user
360                                     <1>
361                                     <1> ; 18/10/2015 - Retro UNIX 386 v1 (local variables for 'namei' and 'sysexec')
362 00018F20 ????????                   <1> pcore:      resd 1  ; physical start address of user's memory space (for sys exec)
363 00018F24 ????????                   <1> ecore:      resd 1  ; physical address of user's stack/last page (for sys exec)
364 00018F28 ????????                   <1> nbase:      resd 1  ; physical base address for 'namei' & 'sysexec'
365                                     <1> ; 23/07/202 - TRDOS 386 Kernel v2.0.5
366                                     <1> ;ncount: resw 1
367 00018F2C ????????                   <1> ncount: resd 1      ; remain byte count in page for 'namei' & 'sysexec'
368                                     <1> ;argc:      resw 1
369 00018F30 ????????                   <1> argc:       resd 1  ; argument count for 'sysexec'
370 00018F34 ????????                   <1> argv:       resd 1  ; argument list (recent) address for 'sysexec'
371                                     <1>
372                                     <1> ; 03/06/2015 - Retro UNIX 386 v1 Beginning
373                                     <1> ; 07/04/2013 - 31/07/2013 - Retro UNIX 8086 v1
374 00018F38 ??                         <1> rw: resb 1 ;; Read/Write sign (iget)
375                                     <1> ; 23/07/2022
376 00018F39 ??????                     <1>     resb 3
377                                     <1>
378                                     <1> alignb 4
379                                     <1>
380                                     <1> ; 24/04/2016
381 00018F3C ????????                   <1> ii: resd 1 ; first cluster of the program file
382 00018F40 ????????                   <1> i.size:     resd 1 ; size of the program file
4008
4009                                        alignb 4
4010
4011                                    ; 23/05/2016 (TRDOS 386)
4012                                    ; 14/10/2015 (Retro UNIX 386 v1, 'unix386.s')
4013 00018F44 ????????                  cr3reg:      resd 1  ; cr3 register content at the beginning of the timer
4014                                                         ; (or RTC) interrupt handler.
4015
4016                                    ; 10/12/2016 (callback)
4017                                    ; 10/06/2016
4018                                    ; 19/05/2016
4019                                    ; 18/05/2016 - TRDOS 386 feature only !
4020 00018F48 <res 100h>                timer_set: resd 16*4   ; 256 bytes memory space for 16 timer events
4021                                        ; Timer Event Structure: (max. 16 timer events, 16*16 bytes)
4022                                        ;       Owner:        resb 1 ; 0 = free
4023                                        ;                            ;>0 = process number (u.uno)
4024                                        ;       Callback:     resb 1 ; 0 = response byte address (phy)
4025                                        ;                             1 = callback address (virtual)
4026                                        ;       Interrupt:    resb 1 ; 0 = Timer interrupt (or none)
4027                                        ;                            ; 1 = Real Time Clock interrupt
4028                                        ;       Response:     resb 1 ; 0 to 255, signal return value
4029                                        ;       Count Limit:  resd 1 ; count of ticks (total/set)
4030                                        ;       Current Count:       resd 1 ; count of ticks (current)
4031                                        ;       Response Addr: resd 1 ; response byte (pointer) address
4032                                        ;                            ; (or callback -user service- address)
4033
4034
4035                                    ; 17/04/2021
4036                                    ; (memory page swap parameters are disabled as temporary)
4037                                    ;
4038                                    ; Memory (swap) Data (11/03/2015)
4039                                    ; 09/03/2015
4040                                    ;swpq_count: resw 1 ; count of pages on the swap queue
4041                                    ;swp_drv:    resd 1 ; logical drive description table address of the swap drive/disk
4042                                    ;swpd_size:  resd 1 ; size of swap drive/disk (volume) in sectors (512 bytes).
4043                                    ;swpd_free:  resd 1 ; free page blocks (4096 bytes) on swap disk/drive (logical)
4044                                    ;swpd_next:  resd 1 ; next free page block
4045                                    ;swpd_last:  resd 1 ; last swap page block
4046
```

```
                                    alignb 4
                                
                                    ; 10/07/2015
                                    ; 28/08/2014
00019048 ????????                   error_code: resd 1
                                    ; 29/08/2014
0001904C ????????                   FaultOffset:       resd 1
                                    ; 21/09/2015
00019050 ????????                   PF_Count:   resd 1  ; total page fault count
                                                        ; (for debugging - page fault analyze)
                                                        ; 'page_fault_handler' (memory.inc)
                                                        ; 'sysgeterr' (u9.s)
                                
                                    ; 29/04/2016 (TRDOS 386 = TRDOS v2.0)
                                    ; 22/08/2015 (Retro UNIX 386 v1)
                                    buffer:
00019054 ????????????????               resb    8
                                    readi_buffer:
0001905C <res 200h>                     resb    512
0001925C ????????????????              resb    8
                                    writei_buffer:
00019264 <res 200h>                     resb    512
                                    ; 24/10/2016
00019464 ????????????????              resb    8
                                    rw_buffer:
0001946C <res 800h>                     resb    2048  ; general purposed, r/w sector buffer
                                
                                    %if 1
                                    ; 17/01/2021
00019C6C <res 80h>                  edid_info:  resb 128 ; VESA EDID (monitor capabilities) info
                                    ; 28/11/2020
00019CEC ??                         pmi32:              resb 1  ; (>0) use VESA VBE3 protected mode calls
00019CED ??                         vbe_mode_x: resb 1  ; VESA VBE3 video bios mode set options
00019CEE ????                       video_mode: resw 1  ; VESA VBE3 video mode (with option flags)
                                    ; 30/11/2020
00019CF0 ????????                   vbe3bios_addr:      resd 1  ; new (writable mem) address of VBE3 bios
                                    ; 02/12/2020
00019CF4 ????????                   pmid_addr:  resd 1  ; PMInfoBlock ('PMID') linear address
                                    ; 14/01/2021
                                    ; 06/12/2020              ; VESA VBE 3 video state
                                    ;vbe3stbufsize: resw 1    ; video regs/dac/bios state buffer size
                                    ;                         ; block size in bytes
                                    ; 16/01/2021
00019CF8 ????                       vbe3stbsflags: resw 1     ; video regs/dac/bios state buffer size
                                    ;                         ; pointer flags for buffer state options
                                    %endif
                                
                                    %if 1
                                    ; 10/12/2020
                                    LFB_Info:
00019CFA <res 10h>                              resb 16 ; Linear Frame Buffer info block
                                
                                    ;24/11/2020 - TRDOS 386 v2.0.3
                                    ; BOCHS/PLEX86 VESA VBE3 MODE INFO extension to TRDOS 386 v2 kernel
                                    MODE_INFO_LIST:
00019D0A <res 44h>                              resb 68 ; mode + 66 byte VESA vbe3 mode info (4F01h)
                                    %endif
                                
                                    ; 05/01/2021
00019D4E ??                         ufont:              resb 1  ; (VGA graphics) user font flags
                                                        ; bit 7 - permission flag for int 31h
                                                        ; bit 4 - 8x16 user font ready/loaded flag
                                                        ; bit 3 - 8x8 user font ready/loaded flag
                                                        ; bit 1 - select 8x16 user font (sysvideo)
                                                        ; bit 0 - select 8x8 user font (sysvideo)
00019D4F ??                                 resb 1  ; 19/01/2021
                                    ; 17/01/2021
00019D50 ??                         srvsf:              resb 1  ; 'save restore video state' permission flag
                                    ; 18/01/2021
00019D51 ??                         srvso:              resb 1  ; video state buffer save/restore option
00019D52 ????????                   VideoStateID:       resd 1  ; used to verify state saved by same prog
                                    ; 29/01/2021
00019D56 ????                       v_width:    resw 1  ; screen (display page) width
00019D58 ??                         v_ops:              resb 1  ; 'sysvideo' graphics data transfer option
00019D59 ??                         v_bpp:              resb 1  ; bits per pixels ('sysvideo')
00019D5A ????????                   v_mem:              resd 1  ; video memory ('sysvideo')
00019D5E ????????                   v_siz:              resd 1  ; video page size ('sysvideo')
00019D62 ????????                   v_str:              resd 1  ; window start adress ('sysvideo')
00019D66 ????????                   v_end:              resd 1  ; window end (end+1) adress ('sysvideo')
                                    ; 31/01/2021
                                    ; 01/01/2021
                                    ;maskbuff:     ;resd 1    ; user's bitmask buffer addr ('sysvideo')
00019D6A ????????                   maskcolor:  resd 1  ; VGA/SVGA pixel mask color ('sysvideo')
                                    ; 27/02/2021
00019D6E ????????                   pixcount:   resd 1  ; pixel count ('sysvideo' window ops)
                                    ; 02/02/2021
00019D72 ????????????????           buffer8:    resd 2  ; 8 bytes small buffer for 'sysvideo'
                                
                                    bss_end:
                                
                                    ; 27/12/2013
                                    _end:  ; end of kernel code
```